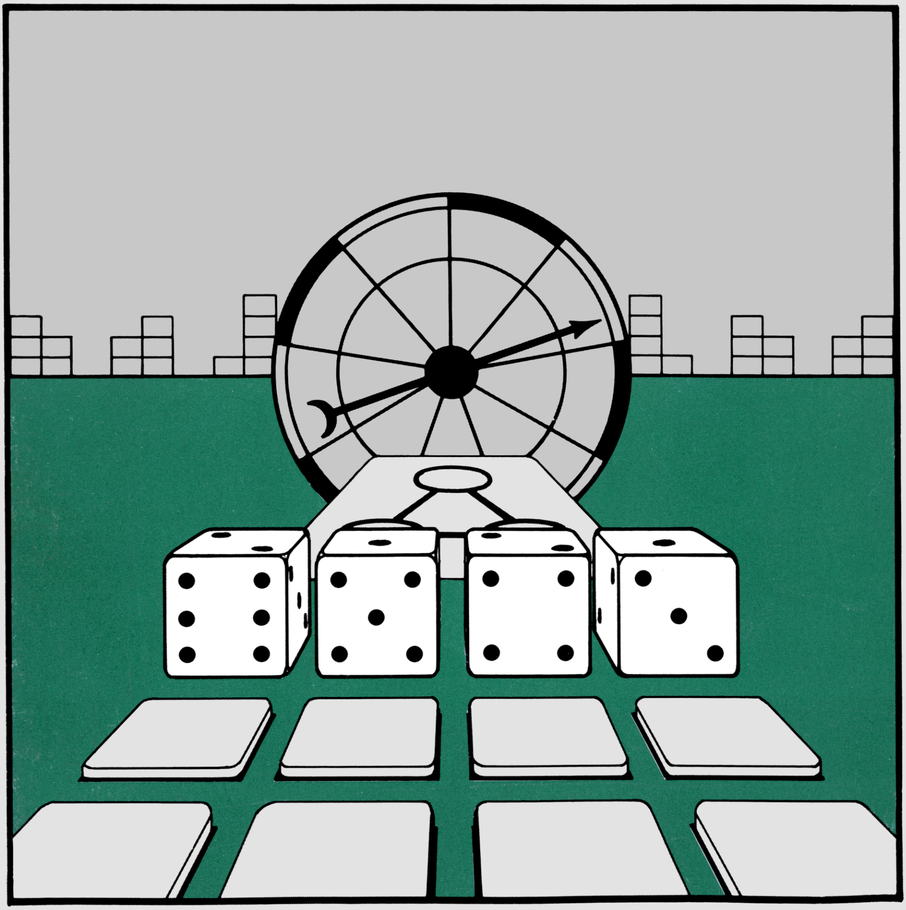
dP Series in Calculators
Number 1

TAKE A CHANCE WITH YOUR CALCULATOR

Probability Problems for Programmable Calculators

Lennart Råde



TAKE A CHANCE
WITH YOUR CALCULATOR

dP Series in Calculators
Number 1

TAKE A CHANCE WITH YOUR CALCULATOR

Probability Problems for Programmable Calculators

Lennart Råde

10 9 8 7 6 5 4 3 2 1

Library of Congress catalog card number: 77-088868

dilithium Press
P. O. Box 92
Forest Grove, Oregon 97116

ISBN: 0-918398-07-X

Printed in the United States of America

CONTENTS

part one

PROBLEMS

part two

COMMENTARIES

part three

PROGRAMS

PREFACE

Programmable calculators will significantly influence the study and teaching of mathematics. These machines are the "slide rules" of today but much more powerful and efficient than slide rules have ever been. Programmable calculators are not only wonderful tools for making numerical calculations; their existence will change completely the way some branches of mathematics are taught. One such field that can be influenced greatly is that of probability, with its applications to statistics and other areas. This is mainly because students, on their own, can simulate random experiments easily and thereby gain concrete knowledge of randomness and stability in such experiments.

This little book is a first step to introduce the use of programmable calculators in the study of probability and statistics. The book can be used as a source of ideas for teachers trying such an approach, or it can be used for self-study as an introduction to these fields. It can also be used as a supplement to existing teaching materials in probability and statistics.

Lennart Råde
Gothenburg, Sweden
January 1977

TO THE READER

In order to enjoy and use this book you must have access to a programmable calculator. But it is not necessary to have a knowledge of probability and statistics. You can let this book serve as your introduction to these fascinating areas of mathematics. It is also not necessary to be expert in programming a calculator. But by working the exercises in this book you will gain ability and assurance in such programming.

Part one of the book consists of 143 exercises, most of which deal with the simulation of some random experiment. Commentaries on these exercises appear in **part two** of the book. Here we give, among other things, hints to help you with the programming, references to literature, historical anecdotes, and so on. Often there is also given a mathematical analysis of the random experiment simulated in the exercise. This analysis is usually given only for a special case of the problem. You should compare the results of this analysis with the results you have obtained when simulating the random experiment. To do so it is of course not necessary for you to understand in detail the mathematical analysis in the commentary. If you have studied probability and statistics, try applying your knowledge in these areas to calculate confidence intervals, do χ^2-tests, and so on. Even if you have studied probability and statistics, it may be that you are not familiar with the method used in some commentaries to find probabilities and expectations (use of conditional probabilities and conditional expectations in combination with flow diagrams). In such cases you can consult Appendix 1 in **part two** where the method is explained in connection with a specific example.

Part three contains programs for Hewlett-Packard 25 and Texas SR56 calculators for all exercises in this book. All simulations in the book deal with discrete random experiments. Continuous problems are not treated.

In many of the exercises you are asked to write a program that will find the mean of a large number of observations but not the variance or the standard deviation of these observations. This has been done to keep the book on an elementary level, without requiring too much technical knowledge in probability theory. But if you are familiar with the concepts of variance and standard deviation, you should extend the programs to include calculation of the standard deviation. You can then also calculate confidence intervals for expectations. It is easy to have the calculator generate so many observations that normal approximation theory can be used.

In many of the exercises a random number generator is used. In the commentaries and in the booklet with programs only the 147-generator is used. This fact should not be interpreted as a recommendation to use only this calculator. Use and compare different calculators so that you can make a decision about what generator you think will give reliable results. You are also urged to make your own random number generator and to investigate its properties with the methods explained.

This book contains a large, but still finite, number of simulations which can be made on a programmable calculator. On your own, you can easily find other challenging simulations. Observe also that by changing one or more of the program steps in a written program, you often can simulate a new random experiment.

Part three of this book contains programs for the exercises. All programs are written for Hewlett-Packard 25 and Texas Instrument SR56 calculators.

A given problem can be programmed in many different ways. It is possible that many programs in this book can be made shorter and more elegant.

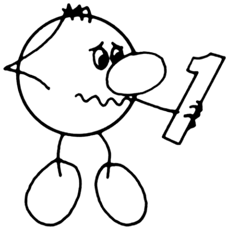
All the program steps to each program are given. Furthermore, instruction is given on how to start the programs, that is, how to store numbers in registers before a program is run. When we say that a program can be repeated, we mean that a program can be rerun directly with the R/S-key. In some cases numbers must be stored in registers, and other registers must be cleared before a program is repeated. This process is also described in most of the programs.

When you are programming a complicated exercise it is often helpful to draw a flow chart first—*before* the program is written. Examples of such flow charts are given in the commentaries to the exercises. If you are running a large number of simulations of a random experiment, it is advisable to write pauses in the program to make easy the check on how far the calculations have proceeded. It is important to check that a program is written correctly and entered correctly into the calculator. It is advisable at the start to make a few simulations to get an idea of how a program works.

In probability theory and statistics, random experiments are studied with the aid of mathematical tools. By using a programmable calculator you can perform such experiments in a direct and challenging way. Have fun when you *take a chance with your calculator.*

part one

PROBLEMS



1. RANDOM DIGITS

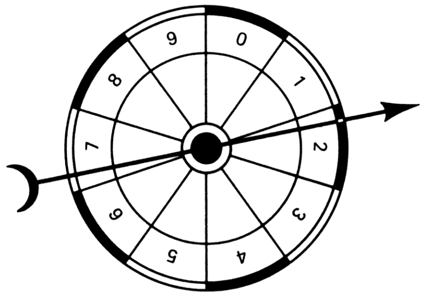


Figure 1

Here is a very simple method which can be used to generate a sequence of random digits, that is, a sequence of outcomes obtained by spinning the spinner in Figure 1. We start with a decimal number x_0 between 0 and 1 and which has at least five decimals (use as many decimals as your calculator allows) e.g.,

$$x_0 = 0.379645937$$

We then generate a new decimal number between 0 and 1 by multiplying x_0 with 147 and taking the fractional part of the product. If in general the fraction part of x is denoted by FRACx or FRAC(x), the new decimal number x_1 is given by the formula

$$x_1 = \mathrm{FRAC}(147x_0)$$

For the case above we obtain

$$x_1 = 0.807952740$$

because $147x_0 = 55.80795274$. Then the first four decimals of x_1 are the first four random digits. In this case these are

$$8 \quad 0 \quad 7 \quad 9$$

Then the same procedure is applied to x_1, that is, we calculate

$$x_2 = \mathrm{FRAC}(147x_1)$$

which is this case gives $x_2 = 0.769052800$. Thus the next four random digits are

$$7 \quad 6 \quad 9 \quad 0$$

Then we continue according to the formula

$$x_{n+1} = \mathrm{FRAC}(147x_n)$$

and each time we keep the first four decimals. In the following exercises, this method of generating random digits is called "the 147-generator."

EXERCISES

1. Continue the sequence started above until you have 400 random digits. Find the number of zeros, the number of ones, and so on, among these 400 random digits.
2. Program your calculator so that it generates random digits with the 147-generator.
3. It is sometimes convenient to generate random digits one by one. This can be done as follows: in the same way as with the 147-generator, you generate decimal numbers x_0, x_1, x_2 . . . and for each x_n you find

   $$\mathrm{INT}(10x_n)$$

   where $\mathrm{INT}(10x_n)$ is the integer part of $10x_n$. This means that for each number x_n you generate only the first decimal. Write a program that generates random digits according to this method.
4. Write a program that generates a large number of random digits (say 10^2 or 10^3 random digits) and finds the number of (or the frequency of) **(a)** zeros and **(b)** fives.
5. Write a program that generates 10^n random digits and stores in five different registers the frequencies for the digits, 0, 1, 2, 3, and 4.
6. Write the program in Exercise 5 in such a way that it can easily be changed to give frequencies also for the digits 5, 6, 7, 8, and 9. By calculating the same sequence of ran-

dom digits two times you can then find the frequencies of all the digits 0, 1, 2, . . . 9. *Hint*: Change a random digit x to $9-x$.

7. Program your calculator to generate 10^n random digits and to find out how many times the generated digit is odd.

2. TESTING A RANDOM DIGIT GENERATOR

It is very important to test whether a random digit generator is satisfactory. This can be done with many different aspects of a generated string of random digits. One simple method is to use the *frequency test*, that is, to find the frequencies of the ten different digits in a generated sequence of digits. For a good random digit generator, these frequencies should be about the same. As an alternative we can calculate the relative frequencies, all of which should be about 0.1.

Suppose that "random digits" are generated as follows. First 100 zeros are generated, then 100 ones, then 100 twos, and so on. Obviously this is not a good random digit generator but it will pass the frequency test.

A more sophisticated test is the *poker test*, which is made as follows. Generate 400 random digits, for example. Group them in order in 100 groups of four digits in each group. These four-digit groups can be classified according to the following table, which also gives probabilities for different possibilities with the assumption that the random digit generator is a perfect one.

Possibility	*Probability*
All different, e.g., 7093	0.504
One pair, e.g., 1731	0.432
Two pairs, e.g., 1771	0.027
Three of a kind, e.g., 8388	0.036
Four of a kind, e.g., 5555	0.001

A good random digit generator should give these possibilities with relative frequencies close to the probabilities in the table.

EXERCISES

8. Check the 147-generator with the frequency test. Use the program in Exercise 6, for example.
9. The commentary to Section 1, Random Digits, suggests that instead of 147 you can also use the following numbers as factors: 83, 117, 123, 133, 163, 173, 187, and 197. Check these random digit generators with the frequency test.
10. In the handbook of a popular programmable calculator, a random digit generator is recommended that is similar to the 147-generator but where the decimal numbers x_0, x_1, $x_2, \ldots$ are generated by the formula

$$x_{n+1} = \text{FRAC}(x_n + \pi)^5$$

    Check this random digit generator with the frequency test also.
11. Work through Exercises 8–10, but in each case use the poker test described above.
12. In connection with the poker test, you can also group in strings of five digits. Then there are the following possibilities and corresponding probabilities for a perfect random digit generator.

Possibility	*Probability*
All different, e.g., 73289	0.3024
One pair, e.g., 23783	0.5040
Two pairs, e.g., 71731	0.1080
Three of a kind, e.g., 55452	0.0720
Full house, e.g., 83838	0.0090
Four of a kind, e.g., 63666	0.0045
Five of a kind, e.g., 77777	0.0001

    Test some random digit generators with the poker test also.
13. Program your calculator to generate 10^n times three random digits with the method described on page 1 (in each decimal number x_n the first three decimals are generated) and to find the frequency for the event for which three digits are different. What is the probability for this event for a perfect random digit generator?

14. Redo Exercise 13 but use instead the method from Exercise 3 to generate random digits. Thus three random digits are obtained by taking the first decimal in three successive numbers x_n.

15. Write a program that 10^n times generates four random digits y_1, y_2, y_3, and y_4 and finds the frequency of the event that

$$y_1 \neq y_2, \quad y_2 \neq y_3, \quad y_3 \neq y_4$$

What is the probability for this event for a perfect random digit generator?

3. TOSSING DICE

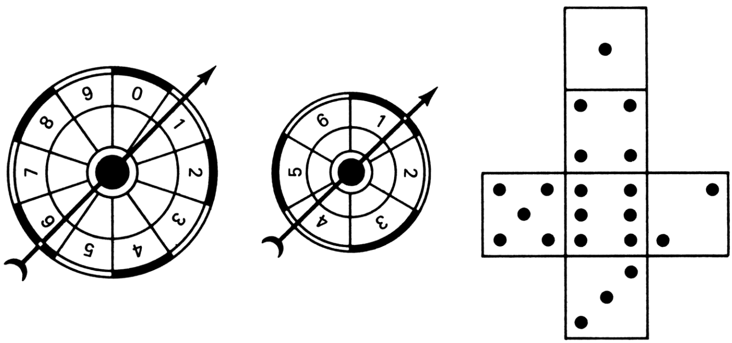


Figure 2

Tosses of a symmetric die can be simulated by generating random digits with the method described in Exercise 3 and disregarding outcomes 0, 7, 8, and 9. However, it is better to use the method that follows. Generate decimal numbers x between 0 and 1 as in connection with the 147-generator, and for each x calculate

$$[6x] + 1$$

where $[6x]$ is the integer part of $6x$. If only $[6x]$ is calculated without adding 1, results of 0, 1, 2, 3, 4, and 5 points are obtained from tossing a die.

EXERCISES

16. Make successive tosses of a die with your calculator by generating random digits with the method described in Exercise 3 and by disregarding the digits 0, 7, 8, and 9.
17. Make successive tosses of a die with your calculator by using random numbers x between 0 and 1 and by calculating $[6x] + 1$.
18. Compare the two methods you have used in the two previous exercises to toss a die. In which case did you get a shorter program? Find which method takes a shorter time to generate 1000 tosses, for example.
19. *Stability of the relative frequencies.* Write a program that makes successive tosses of a symmetric die and after each toss shows the relative frequency (in tosses made so far) of each of the following events:
    (a) The toss gives three points;
    (b) The toss gives at most two points;
    (c) The toss gives an odd number of points.
20. Write a program that tosses a symmetric die a number of times and makes it possible to find at the end of the sequence of tosses the frequencies for six different possible outcomes.
21. Program your calculator to simulate a number of tosses of a symmetric die and to find the mean $\overline{x}$ of the number of points obtained in the tosses.
22. Study the random variation of the mean $\overline{x}$ of the number of points obtained in ten tosses of a symmetric die. Make ten tosses 100 times, for example, and draw a histogram showing the distribution of the means obtained.
23. Program your calculator to simulate a number of tosses of a symmetric die and to give the standard deviation s (or the variance s^2) of the number of points obtained in the tosses.
24. Redo Exercise 22 but study the random variation of the standard deviation s in ten tosses of the die.
25. *Tossing two dice.* Write a program that makes successive tosses of two symmetric dice and which, after each such toss, shows the relative frequency for the event that the sum of the number of points obtained on the two dice is larger than 7.

26. Write a program that makes successive tosses of two symmetric dice. Let A be the event that the second die gives more points than the first. Write the program so that after each ten tosses the calculator first shows the frequency of A in the last ten tosses, and then the relative frequency of the event A in all the tosses made so far.
27. Repeat Exercise 26 but in this case let A be the event that at least one of the dice gives three points.
28. *Waiting for a six.* Program your calculator to make successive tosses of a symmetric die until a six is obtained. The program should repeat this experiment a number of times and at the end calculate the mean $\overline{x}$ and the standard deviation s of the different number of tosses.
29. Write a program that makes successive tosses of two symmetric dice and each time notes the larger of the two numbers of points obtained. The program should also find the mean of the larger numbers in the trials.
30. Redo Exercise 29 but so that you obtain the mean $\overline{x}$ of the larger number in 10^n tosses of m symmetric dice. Study how the mean depends on the value of m.
31. Play the following game between two players on your calculator. Player A tosses a die and then Player B is also to toss a die. But first Player B must predict if her score will be more or less than A's score. If B predicts correctly, she wins; otherwise A wins. What is the optimal strategy of Player B? Estimate by simulation the probability that B will win, if she uses the optimal strategy. As an alternative you can write a program that allows you as Player B to play against the calculator, which acts as Player A. You can also let the calculator keep track of how many times you have won.

4. THE ART OF SIMULATING SPINNERS

In the preceding sections we have simulated spins of the spinners in Figure 3.

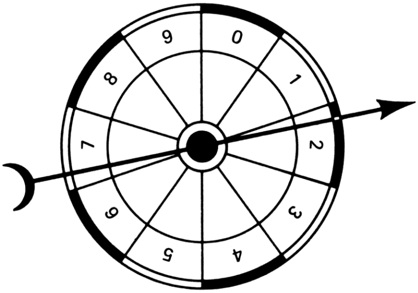


Figure 3

Now consider the spinner in Figure 4. It gives the outcomes 0 and 1, each with probability 1/2.

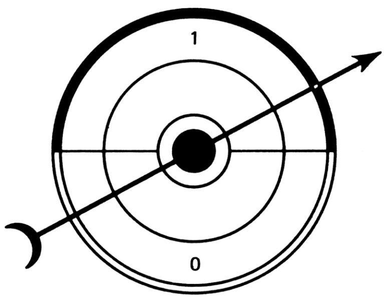


Figure 4

This spinner can be simulated by the following method. Generate random numbers x_n between 0 and 1 as done with the 147-generator and calculate for each x_n the integer part of $2x_n$; that is, find

$$[2x_n]$$

Another possibility is to calculate

$$[x_n + 0.5]$$

the integer part of $x_n + 0.5$.

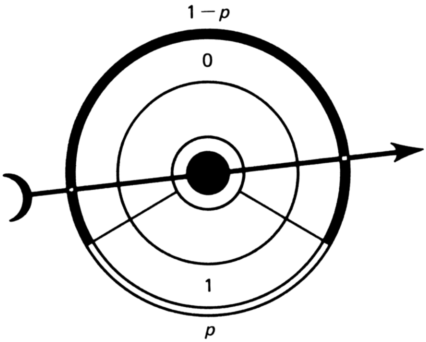


Figure 5

Now consider the spinner in Figure 5. To simulate this spinner we calculate

$$[x_n + p]$$

This gives trials of an experiment such that the outcome 1 occurs with probability p and the outcome 0 occurs with probability $1 - p$. This is a very important experiment.

EXERCISES

32. Write a program that makes successive spins of the spinner in Figure 4. Use the formulas **(a)** $[2x_n]$ and **(b)** $[x_n + 0.5]$.
33. Simulate 10^n spins of the spinner in Figure 5 with $p = 0.6$. Find the number of ones obtained.
34. *Wait for one.* Consider the spinner in Figure 5 with $p = 0.2$. Write a program that repeats a number of times the experiment of spinning the spinner until the outcome 1 is obtained. The program should also give the mean $\overline{x}$ and the standard deviation s of the number of spins in the 10^n trials.
35. Study the random variation of the number of ones obtained in 10 spins of the spinner in Figure 5 with $p = 0.4$. Make ten spins (with your calculator of course) 10^n times, for example, and draw a bar diagram that shows the distribution of the number of ones obtained.

36. Write a program that 10^n times spins the spinner in Figure 5 twice 10^n times and in one register stores the frequency of the outcome 00 (both spins give 0) and in another register stores the frequency of the event that the outcome is 01 or 10. Use **(a)** $p = 0.5$ and **(b)** $p = 0.6$.

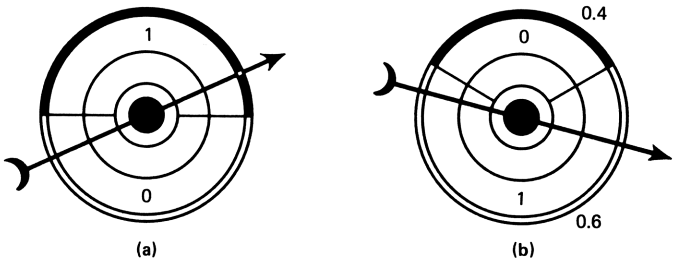


Figure 6

37. Consider the experiment whereby Spinners A and B in Figure 6 are spun simultaneously. Estimate by simulation the probability that Spinner A produces the outcome 1 before Spinner B.

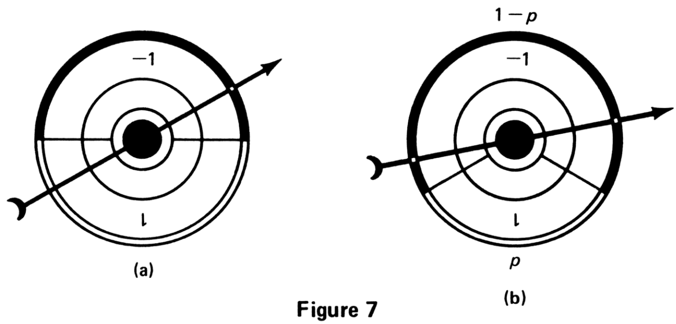


Figure 7

38. Write programs that simulate successive spins of the spinner in **(a)** Figure 7(a) and **(b)** Figure 7(b).

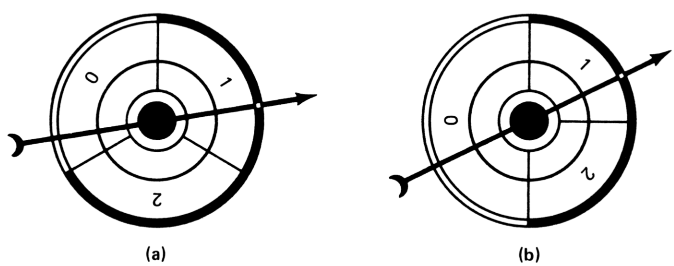


Figure 8

39. Write a program that simulates successive spins of the spinner in **(a)** Figure 8(a) and **(b)** Figure 8(b). The programs should also store the frequencies of the different outcomes in three registers.

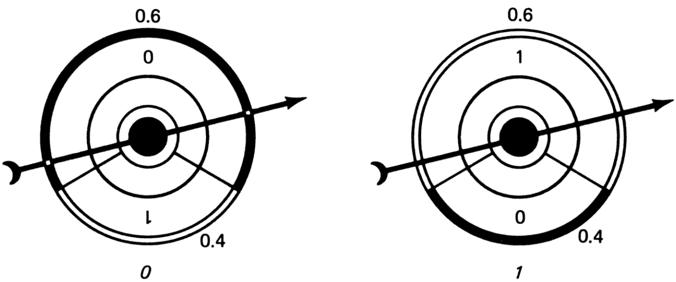


Figure 9

40. Consider the spinners in Figure 9. Observe that the spinners have been numbered *0* and *1*. Write a program that starts by spinning Spinner *0* and which subsequently lets the number in a spin determine the number of the next spinner to be spun. Thus after outcome 0 Spinner *0* is spun, and after outcome 1 Spinner *1* is spun.

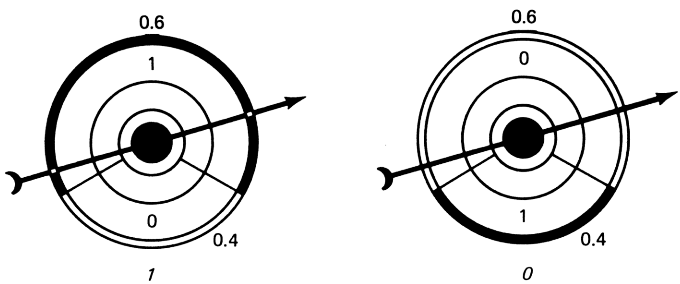


Figure 10

41. Redo Exercise 40 but this time with the spinners in Figure 10. Compare the results from, say, 100 spins of the spinners in Figure 9 with the results from 100 spins of the spinners in Figure 10.

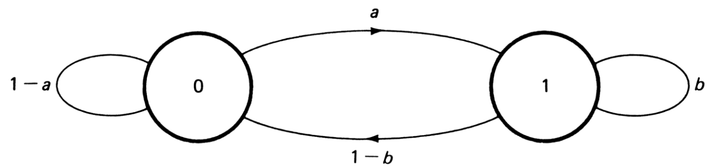


Figure 11

42. *A Markov chain.* Write a program that simulates the Markov chain described by the arrow diagram in Figure 11. Following an outcome 0, the probability of obtaining outcome 1 is a (and that of outcome 0 is $1 - a$). Following an outcome of 1, the probability of obtaining outcome 1 is b (and that of outcome 0 is $1 - b$).
43. Redo Exercise 42 but this time use the formula

$$y_{n+1} = (1 - y_n)\,[x_n + a] + y_n\,[x_n + b]$$

Here the x_n random numbers between 0 and 1 are generated by the method used in the 147-generator.

5. SOME PROBABILITY PROBLEMS

44. Consider the experiment of generating three random digits. Estimate by simulation the probability that the second digit represents a number larger than both the two numbers represented by the other digits.
45. A symmetric die is tossed six times. Estimate by simulation the probability that in at least one toss the number of points obtained coincides with the number of the toss. This event occurs if, for example, the third toss gives three points.
46. Write a program that 10^n times simulates the experiment of tossing a symmetric die six times. The program should store in three different registers the number of times that the number of coincidences between the number of a toss and the number obtained is 0, 1, and greater than 1.

47. Consider a random experiment in which each outcome 1, 2, . . . , n occurs with probability $1/n$. Repeat the experiment n times and let p_n be the probability that in at least one trial the trial number coincides with the number obtained. Estimate p_n by simulation for some values of n, e.g., $n =$ 10, 20, and 100. Note that p_6 was estimated in Exercise 45.
48. Write a program that makes 10^m trials of the random experiment in Exercise 47 and gives in different registers the frequencies for no coincidence, for one coincidence, and for more than one coincidence.
49. *Drawing without replacement.* An urn contains $m + n$ marbles of which m are black and n are white. Take at random, without replacement, r marbles. Study by simulation the random variation of the number of white marbles in the sample of r marbles for **(a)** $m = n = 12$, and $r = 5$; and **(b)** $m = 80$, $n = 40$, and $r = 10$.
50. Write a program that makes 10^s trials of the random experiment in Exercise 49 and calculates the mean of the numbers of white marbles obtained in the trials.
51. *A waiting time problem.* Consider again the urn in Exercise 49. Marbles are taken at random, without replacement, out of the urn until a white marble is obtained. Study by simulation the random variable of the number of marbles taken out of the urn to obtain a white marble.
52. Write a program that makes 10^s trials of the experiment in Exercise 51 and calculates the mean of the number of marbles taken out of the urn. Investigate especially the cases $m = n$ and $n = 1$. Try to guess (from the results of the simulation) formulas for the corresponding expectations. Don't be too quick to look in the commentaries to examine the formulas.
53. Once again consider the urn of Exercise 49. Marbles are taken out of the urn at random, without replacement, until all the n white marbles have been removed. Study by simulation the random variation of the number of marbles taken out of the urn for the cases **(a)** $m = n = 3$ and **(b)** $m = 8$, $n = 4$.
54. Write a program that makes 10^s trials of the random experiments in Exercise 53 and calculates the mean of the marbles taken from the urn.

6. BUILDING AND DESTROYING TOWERS

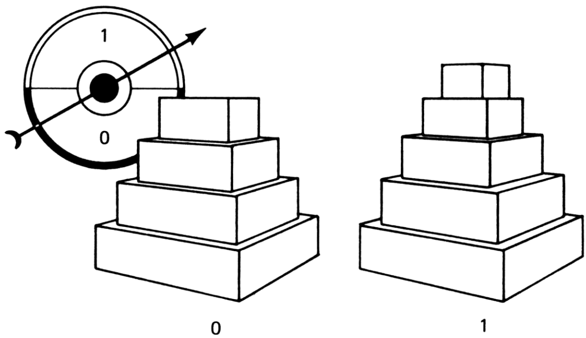


Figure 12

55. Two towers are built of m and n blocks stacked on top of each other. A tower is chosen at random and the uppermost block is taken away from this tower. The procedure is continued until all blocks have been taken from one of the towers. Study by simulation, for some different values of m and n, the random variation of the total number of blocks taken from the towers.
56. Write a program that simulates 10^s trials of the random experiment in Exercise 55, and finds the mean of the total number of blocks taken from the towers.
57. Consider two towers with m and n blocks as in Exercise 55, with $m > n$. As before, a tower is chosen at random and a block is removed from this tower; the procedure is repeated until one of the towers has been destroyed completely. Estimate by simulation for some values of m and n the probability that the smaller tower is destroyed. Study especially the case in which $m = 10$ and $n = 9, 8, 7, \ldots 2, 1$.
58. Two towers are built of m and n blocks. A tower is chosen at random and a block is moved from this tower and put on top of the other tower. The procedure is continued until all the blocks have been moved from one tower to the other. Study by simulation, for some different values of m and n, the random variation of the total number of times a block has been moved from one tower to the other tower.

59. Write a program that makes 10^s trials of the experiment in Exercise 58 and calculates the mean of the total numbers of moves.
60. Consider the random experiment in Exercise 58 with $m > n$. Estimate by simulation, for some different values of m and n, the probability that the smaller tower is destroyed.
61. *Three towers.* Three towers are built of m, n, and r blocks. A tower is chosen at random and a block is taken from the tower. This procedure is continued until one of the three towers has been destroyed. Study by simulation, for some values of m, n, and r, the random variation of the total number of blocks taken from the towers.
62. Write a program that makes 10^s trials of the experiment in Exercise 61 and calculates the mean of the total number of blocks taken from the towers.
63. Three towers are built of 8, 6, and 4 blocks. As in the previous exercises, a tower is chosen at random and a block is removed; this procedure is continued until one of the towers is destroyed. Study by simulation the dependence of the total number of blocks taken from the towers upon the number of blocks remaining at the end in the tower that originally had 8 blocks. Repeat the experiment a number of times and draw a two-dimensional scatter diagram showing the observed values of the two variables.
64. Three towers are built of m, n, and r blocks. A tower is chosen at random and a block is moved from this tower to *another* tower, which also is chosen at random. The procedure is continued until one of the three towers is empty. Study by simulation the random variation of the total number of moves.
65. Redo the investigation in Exercise 64 but assume that when a block is put back on a tower, *this* tower is chosen at random from the three towers. The implication is that a block may be put back on top of the tower from which it was taken.
66. Write a program that makes 10^s trials of the random experiment in Exercise 65 and calculates the mean of the total number of moves.
67. Redo some of the exercises above, but rather than choosing a *tower* at random each time, choose instead a *block* at random from among all the blocks in the tower at that time.

7. TOWER GAMES

Figure 13

68. *The game of two towers.* Two players, *1* and *2*, play the following game. Player *1* has a set of blue blocks and Player *2* a set of red blocks. Player *1* starts by choosing at random one of the numbers 0 or 1 (e.g., with the aid of a coin, a die, or a spinner). If she gets 0 she places a blue block at location 0 in Figure 13; if she gets 1 she places a blue block at location 1. Player *2* repeats the procedure, then Player *1*, and so on. They continue placing their blocks on top of previously placed blocks, so that two towers are growing at locations 0 and 1. A player wins when, for the first time, the uppermost blocks on the two towers both show that player's color.

    Write a program that plays this game, and at the end shows which player was the winner and also shows the total number of blocks placed at locations 0 and 1 during the game. The total number of blocks is the *duration* of the game. Is this a fair game?

69. Write a program that plays the game in Exercise 68 10^n times and at the end of the game gives the number of times Player *1* won and also gives the average duration of the game.

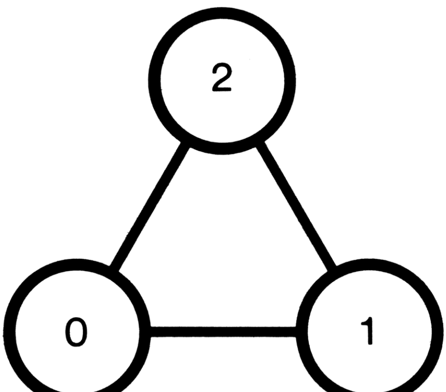


Figure 14

70. Consider a game similar to the game in Exercise 68, but where two players build three towers instead of two; that is, each time the players choose at random from among three towers numbered 0, 1, and 2. As before, a player wins when the uppermost blocks on the three towers all show that player's color.

    Write a program that plays this game and at the end of the game determines which player won and also shows the duration of the game. Is this a fair game?
71. Estimate by simulation the probabilities for different players to win the game described in Exercise 70.
72. Write a program that plays the game in Exercise 70 10^n times and gives the average duration of the games.
73. Simulate a game similar to the games above but this time where two players are putting blocks at random on four towers.
74. *Two towers and three players.* Consider a game in which three players are placing colored blocks at random on two different towers. Each player uses a set of blocks, all one color, but each set is a *different* color. A player wins when the uppermost blocks on the two towers are that player's color. Simulate this game on your calculator.
75. Estimate by simulation the winning probabilities for three players in the game in Exercise 74.
76. Write a program that plays the game in Exercise 74 10^n times and calculates the average duration of the games played.
77. *Two towers and* n *players.* Try to simulate a game similar to the games in Exercise 74 but in which more than three players participate.
78. *Three towers and three players.* Study by simulation a game in which three players are placing blocks on three different towers.

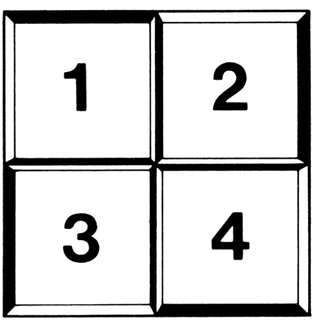


Figure 15

79. *Random tick-tack-toe.* Write a program that plays the following simple kind of tick-tack-toe. Two players are placing blocks at random at the four positions in Figure 15. One player has blue blocks and one has red. A player wins when the uppermost blocks on the towers in a diagonal are that player's color.

8. RUNS AND OTHER PATTERNS

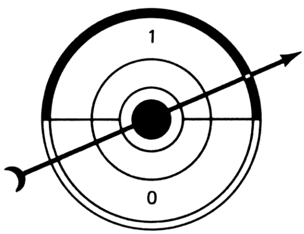


Figure 16

Successive spins of the spinner in Figure 16 will give a random sequence of zeros and ones. If three successive spins all have yielded the outcome 1, a *run* of 1s of length three has occurred. The following exercises all deal with the occurrence of runs and other patterns in a random sequence.

EXERCISES

80. Study by simulation the random variation of the waiting time for a run of outcome 1 of length k when the spinner in Figure 16 is spun; that is, write a program that spins the spinner in Figure 16 until a run of 1s of length k is obtained and at the end gives the total number of spins.
81. Write a program that makes 10^n trials of the experiment in Exercise 80 and calculates the mean and the standard deviation of the waiting times. Study by simulation how these times depend upon k.

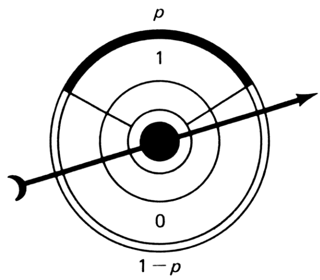


Figure 17

82. Redo Exercise 80 but use the spinner in Figure 17.
83. Redo Exercise 81 but use the spinner in Figure 17.

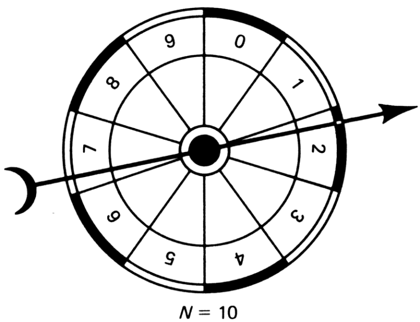


Figure 18

84. Consider successive spins of an N-spinner that give the outcomes $0, 1, 2, \ldots, N-1$ all with probability $1/N$. Study by simulation the random variation of the waiting time until k successive spins have given the same outcome.
85. Write a program that makes 10^s trials of the experiment in Exercise 84 and calculates the mean of the waiting times.
86. Consider the sequence

$$1 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1$$

It has five runs of successive equal elements. Study by simulation the random variation of the number of runs obtained when the spinner in Figure 18 is spun n times.

87. Consider the experiment of making n spins of the spinner in Figure 18 and counting the number of runs in the sequence obtained. Write a program that makes 10^s trials of this experiment and gives the average number of runs.
88. Redo Exercise 86 but use the spinner in Figure 17.
89. Redo Exercise 87 but use the spinner in Figure 17.
90. Consider spins of an N-spinner as in Exercise 84. Study by simulation the waiting time until for the first time two successive outcomes x and y are such that $x < y$; that is, a generated digit is followed by a larger digit.
91. Write a program that makes 10^s trials of the experiment in Exercise 90 and gives the mean $\overline{x}$ and the standard deviation s of the waiting times for patterns xy with $x < y$.
92. Generalize Exercises 90 and 91 to deal with the waiting times until for the first time k successive outcomes x_1, $x_2, \ldots, x_k$ are such that

$$x_1 < x_2 < x_3 < \ldots < x_k$$

93. Consider again spinnings of an N-spinner as in Exercise 90. Study by simulation the waiting times until for the first time two successive outcomes of x and y are such that $y = x + 1$.
94. Redo Exercise 93 but use $y = x + k$ instead of $y = x + 1$.
95. Let q_n be the probability that no runs of 1s of length three are obtained in n spins of the spinner in Figure 17. We can then prove that

$$q_n = 0.5q_{n-1} + 0.25q_{n-2} + 0.125q_{n-3};$$

$$q_0 = q_1 = q_2 = 1$$

Use these formulas to calculate $q_3, q_4, \ldots, q_{30}$ and compare the results with that obtained by using the following approximation

$$q_n \approx \frac{1.236840}{1.0873778^n}$$

96. *Palindromes.* A palindrome from the alphabet $\{0, 1\}$ is a "word" that is the same when its digits are reversed. The following are examples of such palindromes

00 101 011110 010010

Consider the experiment of spinning the spinner in Figure 17 until a palindrome is obtained for the first time. Study by simulation the random variation of the number of symbols produced when for the first time the outcomes constitute a palindrome.

97. Write a program that makes 10^n trials of the experiment in Exercise 96 and gives the mean of the waiting times for a palindrome.

9. RANDOM WALKS

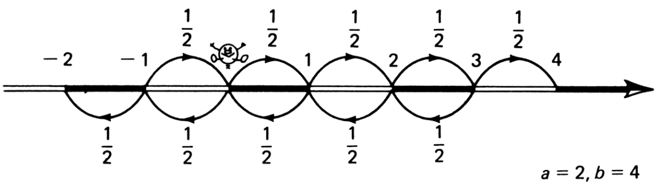


Figure 19

98. A particle performs a random walk between adjacent integer points on the number line as follows. The particle starts at the origin and jumps to Point 1 or Point −1 with probability 1/2 for either jump. Then the particle jumps one step to the right or one step to the left with probabilities 1/2 for either possibility. The particle continues to jump until it reaches Point $-a$ or b. Then the random walk is over. The particle is absorbed in these points. Study by simulation the random variation of the number of steps until absorption and how often the particle is absorbed in $-a$ and in b for some values of a and b.

99. Write a program that makes 10^n trials of the experiment in Exercise 98 and at the end gives the average duration of the random walks and also how many times the particle was absorbed in Point b.

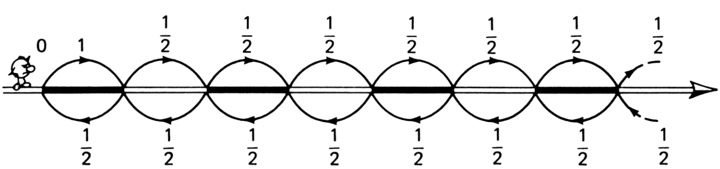


Figure 20

100. Consider the following random walk. A particle starts at the origin and takes a step to Point 1. It then jumps as in Exercise 98 one step to the right or to the left with probability 1/2 for each direction. The random walk is finished when the particle returns to the origin. Study by simulation the random variation of the duration of such walks.

101. Write a program that performs 10^n random walks as described in Exercise 100 and calculates the mean $\overline{x}$ and the standard deviation s of the durations of the walks.

102. Redo Exercises 98 and 99 but use the assumption that the particle jumps to the right with probability p and to the left with probability $1 - p$. Study the effect of choosing different values for p.

103. *Random walk without restrictions.* A particle performs a random walk as in Exercise 98 but with no absorbing barriers. Study by simulation the random variation of the number of returns to the origin in the first $2n$ jumps for some different values of n.

104. Write a program that performs 10^s random walks as described in Exercise 103 and calculates the mean of the number of returns to the origin.

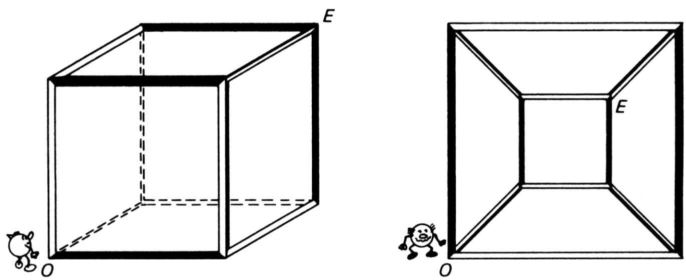


Figure 21

105. *Random walk on a cube.* A particle performs a random walk on a cube as follows. It starts in O (see Figure 21) and jumps to one of the three adjacent corners with probability 1/3 for each. The particle then jumps again to one of the three new adjacent corners. The particle is absorbed when it jumps to E. Study by simulation the random variation of the duration of such random walks.

106. Write a program that performs 10^n random walks as described in Exercise 105 and calculates the average duration of the walks.

107. *Random walk on four-dimensional cube.* Redo Exercise 105 but use the assumption that the random walk takes place on a four-dimensional cube.

108. *The Ehrenfest Diffusion model.* A container has three marbles; another container is empty. One of the marbles is chosen at random and moved to the empty container. Then *one of the three marbles is taken at random* and moved to the other container. The process is continued until all marbles have been moved to the originally empty container. Study by simulation the random variation of the number of moves until this happens.

109. *Complete random walk on square.* A particle makes a random walk between the corners of a square. It jumps each time to one of the two adjacent corners with probability 1/2 for each. The random walk is finished when the particle has jumped to all four corners. Study by simulation the random variation of the duration of such walks.

110. Write a program that performs 10^n random walks as described in Exercise 109 and calculates the average duration of the walks.

111. Consider a random walk as described in Exercise 100 but with an absorbing barrier at Point a, $a > 0$. Simulate such random walks in the visible x-register on your calculator such that the digit 8 performs walks against a background of digits 1. Use the scientific notation of your calculator so that the accumulated duration of the walk at times is displayed to the right in the x-register. At the end the calculator should give the final duration until absorption.

10. SOME STATISTICAL APPLICATIONS

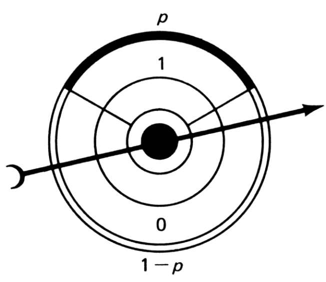


Figure 22

112. *Estimation of an unknown probability.* Program your calculator to make n spins of the spinner in Figure 22 with a probability p, which is unknown to you. The program should then estimate p with the estimate $\hat{p}$, where $\hat{p}$ = the relative frequency of the outcome 1 in n spins.

Compare p and $\hat{p}$. Investigate the accuracy of the estimate $\hat{p}$ by using $n = 10$, $n = 100$, and $n = 1000$, for example. The commentary describes how you can write a program so that the calculator spins a spinner with a probability unknown to you.

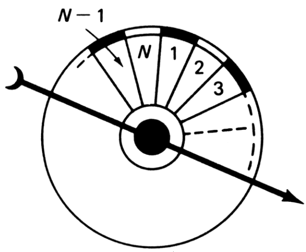


Figure 23

113. *Estimation of an unknown parameter.* Program your calculator to make n spins of the spinner in Figure 23 with a value of N unknown to you. The spinner gives the outcomes $1, 2, 3, \ldots, N$ each with probability $1/N$. The program should also estimate N using the estimate

$$\hat{N}_1 = 2\overline{x}$$

where $\overline{x}$ is the mean of n observations. Investigate as in Exercise 112 the manner in which accuracy of the estimate depends upon n.

114. Redo Exercise 113 but in this case estimate N using

$$\hat{N}_2 = \frac{n+1}{n} x_{\text{max}}$$

where x_{max} is the largest of the n observations.

115. *Consistency of an estimate.* Simulate successive spins of the spinner in Figure 23. In this case you should know the value

of p. Estimate p after each spin with the relative frequency of the outcome 1 in the trials made thus far. Do the estimates approach p as n increases?

116. Simulate successive spins of the spinner in Figure 23 with a value of N which is known to you. Estimate N after each spin using the estimate $2\bar{x} - 1$ (see the commentary to Exercise 113), where $\bar{x}$ is the mean of the observations obtained thus far. Do the estimates approach N as n increases?

117. Redo Exercise 116 but use the estimate $\hat{N}_2$ as defined in Exercise 114.

118. *Efficiency of an estimate.* Study by simulation the random variation of the estimates in Exercises 116 and 117. Repeat the estimation process a number of times (use 10 observations each time, for example) and calculate the variance of the estimates obtained. Which of the estimates seems to have the smaller random variation?

119. *Estimation of confidence level by simulation.* Suppose you make n spins of the spinner in Figure 22 and that you estimate p using the interval estimate

$$p = \hat{p} \pm 2\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

where $\hat{p}$ is the relative frequency as in Exercise 112. Estimate by simulation, for some different values of p and n, the probability that the interval actually contains p.

120. Write a program that spins the spinner in Figure 22 a number of times. The value of p should be unknown to you. The program should also compute the limits for the interval estimate described in Exercise 119. Check to see if the interval contains p. Compare the cases $n = 10$, $n = 100$, and $n = 1000$.

11. MISCELLANEOUS PROBLEMS

121. *Family planning.* Suppose that families in a given society are using a specific method of family planning. They bear children until they have at least one girl and at least one boy. Simulate 10^n families in such a society and find the average number of children in a family and also the proportion of

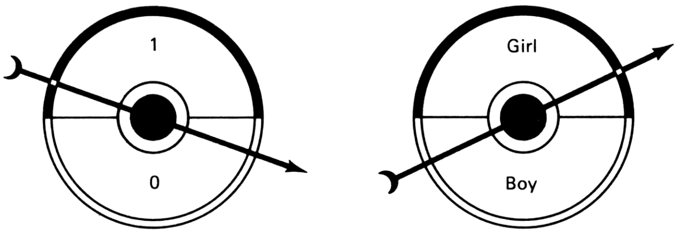


Figure 24

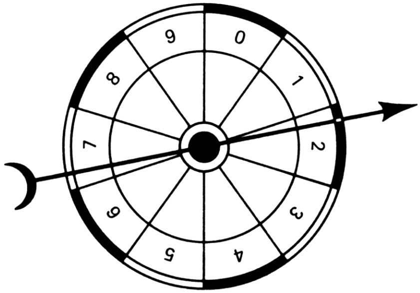
boys among the children in all the families. The probability for bearing a boy is assumed to be 0.5.

122. Redo Exercise 121 but use the assumption that the families bear children until they have at least one girl.
123. Redo Exercise 121 but assume that the families are using the following "stopping rule": at least one girl but at most three children.
124. Redo Exercise 121 using the following stopping rule: at least two girls but at most five children.
125. Redo some of the Exercises 121–124 using the assumption that the probability for bearing a boy is 0.52.
126. *Waiting for 101.* Program your calculator to make successive spins of the spinner in Figure 22 until three successive spins give the pattern 101. The program should also give the number of spins used to produce the pattern.
127. Write a program that makes 10^n trials of the experiment in Exercise 126 and calculates the mean $\overline{x}$ of the number of spins.
128. *Records.* Consider the following sequence of random digits.

$$3 \quad 2 \quad 4 \quad 7 \quad 6 \quad 5 \quad 9 \quad 2 \quad 0 \quad 1$$

Here 3, 4, 7, and 9 are records because these numbers are larger than all previous numbers in the sequence. The first digit is also counted as a record. Program your calculator to generate a sequence of n random digits and to count the number of records.

129. Write a program that 10^m times generates a sequence of n random digits and also calculates the mean of the number of records in the sequences.



$N = 10$

Figure 25

130. *Probability for record.* Consider n spins of the N-spinner in Figure 25, which give the outcomes $0, 1, 2, \ldots, N-1$ with probability $1/N$ for each. Let $P(N, n)$ be the probability that the nth spin gives a record. Write a program that estimates the probability $P(N, n)$. Compare, for example, the cases $N = 10$ and $N = 1000$ for $n = 2, 3, 4$, and so on.
131. *The Birthday Problem.* Consider n spins of the spinner in Figure 25. Let $P(N, n)$ be the probability that at least two of the spins give the same outcome. We can then prove that

$$P(N, n) = 1 - \frac{N(N-1)(N-2)\ldots(N-n+1)}{N^n}$$

Program your calculator to calculate this probability for various values of N and n. Do this especially for $N = 365$. In this case, $P(N, n)$ is the probability that at least two of n persons have the same birthday.
132. Consider the probability $P(N, n)$ from Exercise 131. Program your calculator to find for a given value of N the smallest n such that the inequality $P(N, n) \geqslant 0.5$ holds.
133. Let P_n be the probability that at least one of n persons has the same birthday as you. Find a formula for P_n and use your calculator to determine the smallest value of n for which $P_n \geqslant 0.5$.
134. *Simulation of a ballot.* Suppose that A and B on a ballot have scored a and b votes. The votes are written on pieces of paper and during the counting taken at random from an urn. Program your calculator to simulate counting the votes.

After each vote has been counted the calculator should indicate the number of votes for A and B at that point in the counting.

135. Write a program that simulates a ballot as in Exercise 134 but at the end also indicates the number of times during the counting that A and B have scored the same number of votes.

136. Write a program that simulates 10^n ballots as in Exercise 135 and at the end calculates the average number of times that A and B scored the same number of votes during the countings.

137. Consider the ballot in Exercise 134. Estimate by simulation, and for various values of a and b such that $a > b$, the probability that throughout the counting there are always more votes for A than for B.

138. *Queuing.* A stream of customers arrives at a service facility as follows. The customers arrive at the time epochs 1, 2, 3, 4, . . . , and for each such time epoch the probability is p that a customer will arrive. At the service facility there is one attendant who immediately starts to serve an arriving customer if he is not busy with already arrived customers. The service has the following structure: a service that is going on at a time epoch is ended during the next time interval with probability p_1. Customers arriving when the attendant is busy wait for service in a queue.

Write a program that simulates such a service system. The system is supposed to be empty of customers at time epoch 0, and the program should indicate successively the numbers of customers present in the system at time epochs 1, 2, 3, ... When you simulate the system draw a diagram that illustrates the manner in which the number of customers is changing. Investigate especially what it means for the functioning of the system if $p < p_1$ or $p > p_1$.

139. *Busy period.* A busy period of the service system in Exercise 138 is defined as the time period from the arrival of a customer to an empty system until the time epoch when the system is again empty. Write a program that simulates busy periods of the system and at the end of such a period gives the length of the period and also the number of customers served during the busy period.

140. Write a program that simulates 10^n busy periods and at the end gives the average length of the busy periods and also the average number of customers served during the busy periods.

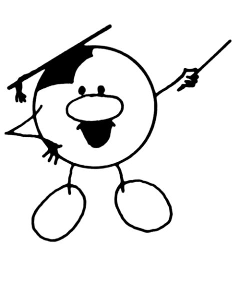
141. *Inverse sampling.* Program your calculator to spin the spinner in Figure 22 until r outcomes 1 have been obtained altogether. The probability p should be unknown to you. Let N be the required number of spins. The program should also calculate the quotient r/N. Decide if r/N is a good estimator of p.

142. Write a program that repeats 10^n times the experiment in Exercise 141, that is, spins the spinner until r outcomes 1 have been obtained and finds r/N. The program should also calculate the mean of all the quotients r/N.

143. Redo Exercise 142 but use $(r - 1)/(N - 1)$ to estimate p instead of r/N. Which of the estimators is better?

part two

COMMENTARIES



1. RANDOM DIGITS

Random digits are extremely important and useful in applied probability. A method of generating random digits on a calculator or a computer is called a *random digit generator*. One such example is the 147-generator. Such generators are, of course, really not random in the same way as the experiments of actually spinning the spinner shown in Figure 1 or tossing a die. When the first number x_0 is chosen, all other numbers are decided upon as well. However, it has been found that the 147-generator gives results similar to those obtained with a spinner similar to the one shown in Figure 1. Sometimes random digits generated by a random digit generator such as the 147-generator are called *pseudorandom digits*.

You are probably curious about the factor 147. You should try other factors as well. You can, however, wait to make such investigations until the next section, which deals with checking random digit generators. Experience, however, indicates that 147 is a suitable factor. Other factors recommended in the literature are 83, 117, 123, 133, 163, 173, 187, and 197.

Observe that the sequence $x_0, x_1, x_2, \ldots$ is periodic. When the number x_0 is repeated the other numbers will be repeated as well. Because there exists a finite number of possible numbers x_n, the number x_0 must sooner or later reappear. By using the factors above you will get periodic sequences with long periods. The author has found that 137 also gives satisfactory results, even if it does not give maximum periods. Not only long periods are important, however. The generated digits should also be "independent." (We will not discuss these matters here.)

The numbers $x_0, x_1, x_2, \ldots$ used in the random digit generators described above also have a random character. We will call them *random numbers between 0 and 1*, and describe them as results of successive random choices of points between 0 and 1 on the number line.

Exercise 5

In order to determine if a generated digit is 0, 1, 2, 3, or 4, you can start by programming to find if the digit is 0, then subtract 1 and determine if the digit is 0, then subtract 1, etc., as illustrated in the flow chart in Figure 26.

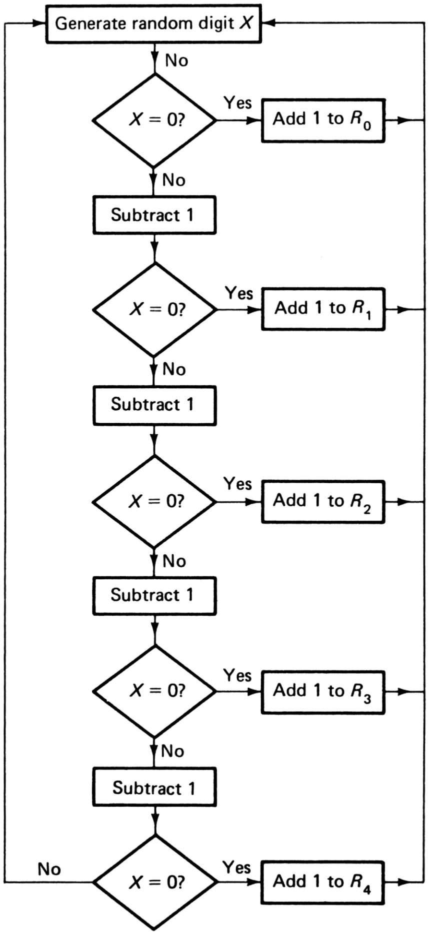


Figure 26

Exercise 6

Another possible way to decide if a generated digit is 0, 1, 2, 3, or 4 is to use the scheme illustrated by the flow chart in Figure 27. In this case you start by determining if the digit is less than 3.

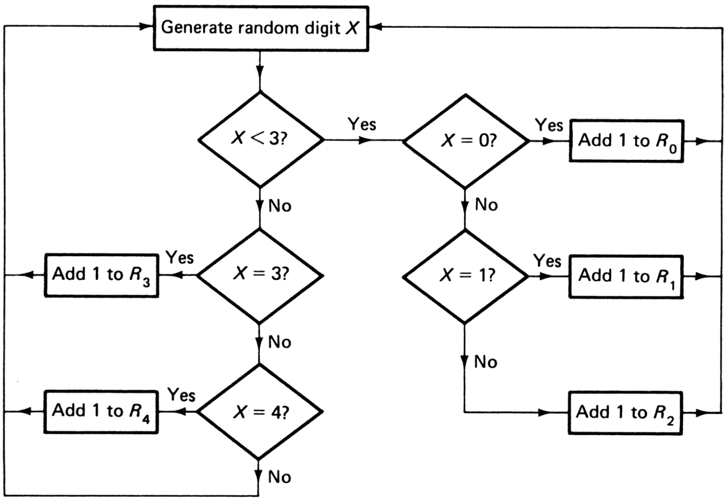


Figure 27

Exercise 7

A suitable method to decide if a digit is even or odd is to divide by 2, then take the fractional part, and finally determine if the fractional part is zero or different from zero.

2. TESTING A RANDOM DIGIT GENERATOR

Your work in this section should give you confidence in some random digit generators so that you will trust results you obtain by using them. We have mainly used the 147-generator above, and also in the programs in the answer book. But this is not to be taken too seriously. You may prefer a factor different from 147, or perhaps a completely different random digit generator. You may read about random digit generators in the book *Random Number Generators* by B. Jansson (Stockholm 1966), or in the paper *Random Number Generators* by T. E. Hull and A. R. Dobell in SIAM Review, vol. 4, 1962. You will find an extensive list of references to random digits in the book *Concepts and Methods in Discrete Event Digital Simulation* by G. S. Fishman (New York, 1973).

We should mention that in addition to frequency and poker tests, many other tests exist for random digit generators. The following table gives you a hint how probabilities for the different possibilities are found in the poker test, where generated random digits are taken four at a time.

Possibility	*Probability*
All different	$\dfrac{10 \cdot 9 \cdot 8 \cdot 7}{10^4}$
One pair	$\dfrac{\binom{4}{2} \cdot 10 \cdot 9 \cdot 8}{10^4}$
Three of a kind	$\dfrac{4 \cdot 10 \cdot 9}{10^4}$
Four of a kind	$\dfrac{10}{10^4}$

The easiest way to find the probability for two pairs is to subtract the sum of the above probabilities from 1. You find in a similar way the probabilities in a poker test in which the digits are taken five at a time.

For an acceptable random digit generator, you should obtain in connection with frequency and poker tests relative frequencies close to the probabilities calculated for a perfect random digit generator. A difficulty is deciding what is meant by "close." The theory of statistics has provided us with simple criteria to use when judging whether observed relative frequencies agree with calculated probabilities. One common method is the χ^2-test. If you know about this test, you should use it in exercises in this book that ask you to compare observed relative frequencies with calculated probabilities.

Exercise 13

The probability is $\dfrac{10 \cdot 9 \cdot 8}{10 \cdot 10 \cdot 10}$ or 0.72

Exercise 15

The probability is $\dfrac{10 \cdot 9 \cdot 9 \cdot 9}{10^4}$ or 0.729

3. TOSSING DICE

Exercises 16–17

The method used in Exercise 16 uses only 60 percent of the generated random digits. In order to obtain 100 tosses of a die you must generate about 166 random digits. The method used in Exercise 16 is thus considerably slower than the method in Exercise 17.

Exercise 20

If the total number of tosses is stored, it is necessary to store only the frequencies for five of the six possible outcomes. With little change you can use the program from Exercise 5.

Exercise 21

If the outcomes 1, 2, . . . , n occur in a random experiment with the probability $1/n$, the expectation (expected value) is given by

$$1 \cdot \frac{1}{n} + 2 \cdot \frac{1}{n} + \ldots + n \cdot \frac{1}{n} = \frac{n+1}{2}$$

For a symmetric die n is 6, and thus the expectation is 3.5.

Exercise 23

If the outcomes 1, 2, . . . , n occur in a random experiment with the same probability $1/n$, we can prove that the variance of the experiment is $(n^2 - 1)/12$, and the standard deviation is $\sqrt{(n^2 - 1)/12}$. For a symmetric die n is 6, which implies that the variance is $35/12 \approx 2.92$, and that the standard deviation is $\sqrt{35/12} \approx 1.71$.

Exercise 25

The probability of obtaining more than 7 is 15/36, or 0.42.

Exercise 28

Consider an event that occurs with probability p in a random experiment. Suppose that independent trials of the experiment are performed until the event occurs. We can then prove that the expected number of trials is $1/p$ and that the standard deviation of the number of trials is $\sqrt{(1-p)/p}$. For a symmetric die p is 1/6, and thus the expected number of tosses in this case is 6, and the standard deviation is $\sqrt{30} \approx 5.48$.

Exercise 29

The expected value of the larger number is

$$1 \cdot \frac{1}{36} + 2 \cdot \frac{3}{36} + 3 \cdot \frac{5}{36} + 4 \cdot \frac{7}{36} + 5 \cdot \frac{9}{36} + 6 \cdot \frac{11}{36} = \frac{161}{36} \approx 4.47$$

Exercise 31

The best strategy for Player B is to guess "more" when Player A scores 1, 2, or 3, and to guess "less" when Player A scores 4, 5, or 6. Her probability to win is thus seen to be 2/3.

4. THE ART OF SIMULATING SPINNERS

Each random experiment with a finite number of outcomes can be interpreted as the spinning of a spinner, which is one reason why it is important to study the random behavior of spinners.

It is obvious why the formula $[2x_n]$ gives 0 and 1 with probability 1/2 for both outcomes. If x_n is between 0 and 0.5, the formula gives 0, and if x_n is between 0.5 and 1, the formula gives 1. Thus both these possibilities are obtained with probability 1/2. The other formulas are proved by similar arguments.

Exercise 34

See the commentary to Exercise 28 for the expectation and standard deviation of the number of spins.

Exercise 35

According to the binomial distribution the probability for x outcomes 1 is given by

$$\binom{10}{x} \cdot 0.4^x \, 0.6^{10-x}$$

The frequencies in, for example, 100 trials are obtained by multiplying these probabilities by 100. Compare the expected frequencies with observed frequencies.

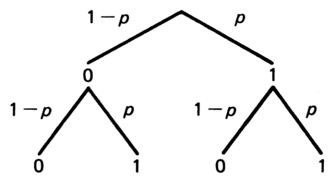


Figure 28

Exercise 36

The probability for 00 is $(1 - p)^2$ and the probability for the event that 01 or 10 occurs is $2p(1 - p)$.

Exercise 37

Suppose that Spinner A gives 1 with probability p_A and that Spinner B gives 1 with probability p_B. The probability that Spinner A gives 1 before Spinner B is then obtained.

$$\sum_{k=1}^{\infty} (1 - p_A)^{k-1} p_A (1 - p_B)^k = \frac{p_A(1 - p_B)}{p_A + p_B - p_A p_B}$$

This probability can also be derived. Call the probability x and condition with respect to outcomes on the first spin of the spinners, which gives

$$x = (1 - p_A)(1 - p_B) \cdot x + p_A(1 - p_B)$$

If, for example, p_A is 0.5 and p_B is 0.6, the probability that A gives 1 before B is 0.25.

Exercise 38

(a) Generate random numbers x_n between 0 and 1 as in the 147-generator and calculate $2[2x_n] - 1$. Another possibility is to use the formula $2[x_n + 0.5] - 1$.
(b) Generate random numbers x_n between 0 and 1 as in the 147-generator and calculate $2[x_n + p] - 1$.

Exercise 39

(a) Generate random numbers x_n between 0 and 1 and calculate $[3x_n]$.
(b) Generate random numbers x_n between 0 and 1 and calculate

$$[x_n + 0.5] + [x_n + 0.25]$$

Observe that this formula for $0 \leqslant x_n < 0.5$ gives 0, for $0.5 \leqslant x_n < 0.75$ gives 1, and for $0.75 \leqslant x_n \leqslant 1$ gives 2.

Exercises 40–43

These exercises are concerned with what is called in probability theory a *Markov chain* with two states, 0 and 1. Typically, a Markov chain constitutes a random experiment with the following simple type of dependence between successive trials: the probabilities in a trial depend upon the outcomes in the previous trial. A Markov chain with states 0 and 1 can, in general, be illustrated by the diagram in Figure 29.

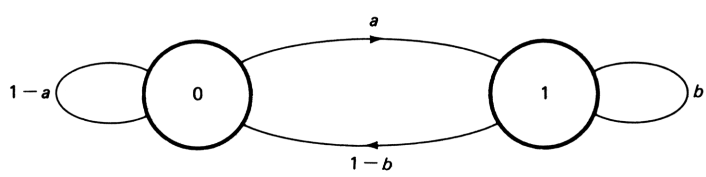


Figure 29

After an outcome 0 the probability for 1 is a, and after an outcome 1 the probability for 1 is b. For $a = b$ we have independent trials of an experiment with outcomes 0 and 1.

Note that in Exercise 40 a has the value 0.4 and b the value 0.6, whereas in Exercise 41 the value of a is 0.6 and the value of b is 0.4.

Following are the results of 50 trials the author has performed with his calculator in connection with Exercises 40 and 41.

Exercise 40:

0 1111111 0 111 0 1 0 111 0
1 00000 11 00 1 00 1111 0 1
0000 11 000000

Exercise 41:

00 1 0 1 0 1 000 11 0 1
0 11 00 1 0 1 0 1 0 11 0
111 00 11 00 11 0 111 0 1
0 11 00

Successive equal outcomes have been combined to emphasize an essential difference between the two sequences of observed outcomes. In Exercise 40 successive outcomes have a tendency to be equal, whereas in Exercise 41 they have a tendency to be different.

A sequence of successive equal observed outcomes is called a *run*. For the two observed sequences of outcomes above, the number of runs are 21 (Exercise 40) and 33 (Exercise 41). We can prove (see L. Råde, *Thinning of Renewal Point Processes*, Gothenburg 1972, p. 139) that the expected number of runs in a sequence of n outcomes generated by a Markov chain as described in Figure 29 is given approximately by

$$1 + 2(n-1)\frac{a(1-b)}{a-b+1}$$

For $n = 50$, $a = 0.4$, and $b = 0.6$ (Exercise 40), the value is 20.6. For $n = 50$, $a = 0.6$, and $b = 0.4$, it is 30.4.

Markov chains are important in probability theory both from a practical and a theoretical point of view. They are named after the outstanding Russian mathematician A. A. Markov (1856–1922) who lived in St. Petersburg. An apocyrphal story about Markov relates a discussion in 1913 about the forthcoming celebration of the Romanov dynasty's 300-year jubilee. Markov is said to have suggested celebrating *instead* the 200-year jubilee of the law of large numbers—one of the cornerstone theorems in probability theory. Markov was also one of the first to suggest that probability theory should be introduced into school mathematics courses.

5. SOME PROBABILITY PROBLEMS

Exercise 44

The probability is

$$\frac{1^2 + 2^2 + 3^2 + 4^2 + 5^2 + 6^2 + 7^2 + 8^2 + 9^2}{1000} = 0.28$$

One can derive this probability by considering the different possibilities for the second digit. For instance, if the second digit is 5, there are 5 possibilities for both the first and the third digit (0, 1, 2, 3, 4) if they are to represent smaller numbers.

Exercise 45

In each toss the probability for a coincidence is 1/6. Then the probability for no coincidences in the six tosses is $(5/6)^6$ and thus the probability for at least one coincidence is given by

$$1 - (5/6)^6 \approx 0.6651$$

Exercise 46

It follows from the commentary to Exercise 45, and the binomial distribution, that the probability for no coincidences is given by

$$(5/6)^5 \approx 0.3349$$

Furthermore the probability for exactly one coincidence is given by

$$\binom{6}{1} \cdot \left(\frac{1}{6}\right) \left(\frac{5}{6}\right)^5 \approx 0.4019$$

The probability for more than one coincidence is then given by

$$1 - 0.3349 - 0.4019 = 0.2632$$

Exercise 47

In each trial the probability for a coincidence is $1/n$. The probability for no coincidences in the n trials is then given by

$$\left(1 - \frac{1}{n}\right)^n$$

and the probability for at least one coincidence is

$$1 - \left(1 - \frac{1}{n}\right)^n$$

which, for increasing n, approaches $1 - e^{-1} \approx 0.6321$. This offers the possibility of estimating the number e by simulation.

Exercise 48

Let q_0 be the probability for no coincidences and q_1 the probability for exactly one coincidence. Then

$$q_0 = \left(1 - \frac{1}{n}\right)^n \quad \text{and} \quad q_1 = \binom{n}{1} \cdot \frac{1}{n} \cdot \left(1 - \frac{1}{n}\right)^{n-1} = \left(1 - \frac{1}{n}\right)^{n-1}$$

Both q_0 and q_1 approach e^{-1} when n increases. We can, more generally, prove that for large n the probability for k coincidences is approximately given by $e^{-1}/k!$, $k = 0, 1, 2, \ldots$ This means that for large n the number of coincidences approximates a Poisson distribution with parameter 1.

Exercise 49

According to the hypergeometric distribution the probability for x white marbles is given by

$$\frac{\binom{n}{x}\binom{m}{r-x}}{\binom{n+m}{r}}$$

The programming of this problem can be made as follows. Store the numbers n and m in different registers. Then choose one of the registers at random in such a way that the probability to choose the register with number n is $n/(n+m)$ and the probability to choose the register with number m is $m/(n+m)$; then subtract 1 from the chosen register. Repeat this procedure but with new numbers now stored in the registers.

Exercise 50

We can prove that the expected number of white marbles is $rn/(n+m)$, which is the expectation of the hypergeometric distribution mentioned in the commentary to Exercise 49.

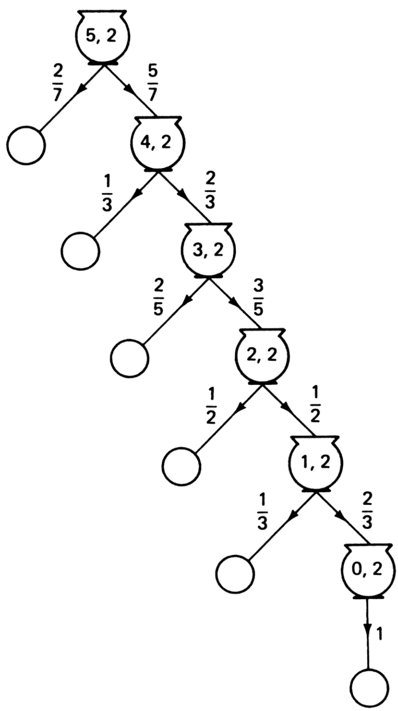


Figure 30

Exercise 51

Figure 30 illustrates the case in which $m = 5$ and $n = 2$. Here "3, 2" means a state when the urn contains 3 black and 2 white marbles.

Let p_k be the probability that in this case k marbles are taken out of the urn. It follows from Figure 31 that

$$p_1 = \frac{2}{7} = \frac{6}{21} \qquad p_2 = \frac{5}{7} \cdot \frac{1}{3} = \frac{5}{21}$$

$$p_3 = \frac{5}{7} \cdot \frac{2}{3} \cdot \frac{2}{5} = \frac{4}{21} \qquad p_4 = \frac{5}{7} \cdot \frac{2}{3} \cdot \frac{3}{5} \cdot \frac{1}{2} = \frac{3}{21}$$

$$p_5 = \frac{5}{7} \cdot \frac{2}{3} \cdot \frac{3}{5} \cdot \frac{1}{2} \cdot \frac{1}{3} = \frac{1}{21} \qquad p_6 = \frac{5}{7} \cdot \frac{2}{3} \cdot \frac{3}{5} \cdot \frac{1}{2} \cdot \frac{2}{3} = \frac{2}{21}$$

Exercise 52

With the aid of a diagram similar to Figure 30 we can prove conditionally that the expected number of marbles taken out of the urn is $(n + m + 1)/(n + 1)$. Thus in the special case $m = n$, the expectation is $1 + [n/(n + 1)]$, and in the special case $n = 1$ the expectation is $1 + (m/2)$.

Exercise 53

The diagram in Figure 31 illustrates the case for which $m = n = 5$, and indicates that it is a laborious task to find probabilities for the various possible numbers of marbles which must be taken from the urn to secure all the white marbles. It might be easier to find these probabilities by simulation.

Exercise 54

Consider the case in which the urn contains three black and two white marbles. The expected number of marbles taken from the urn can then be found with the aid of the diagram in Figure 32.

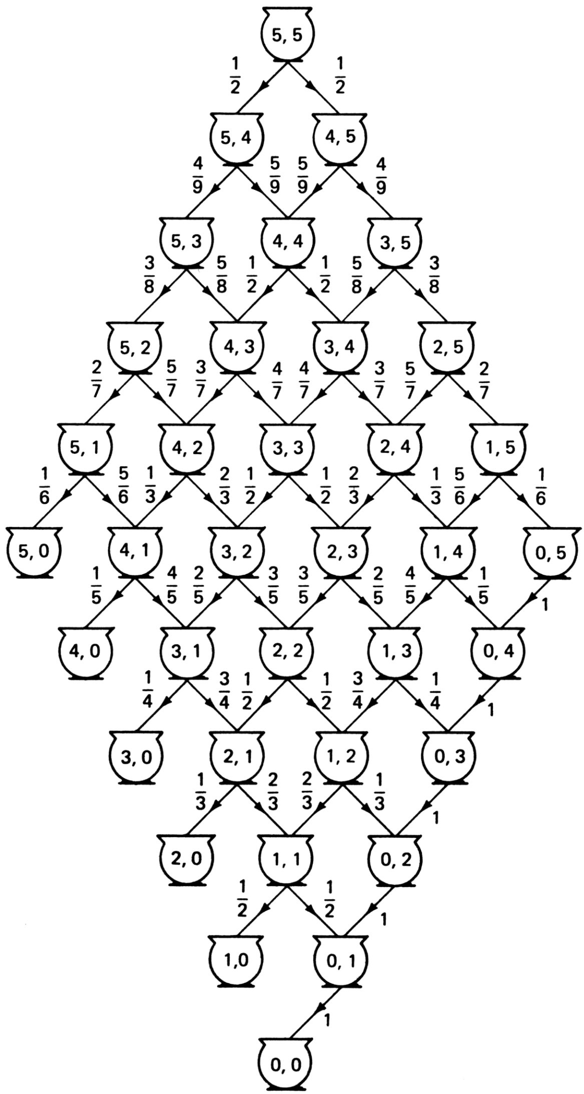


Figure 31

Let x_i be the expectations from the different possible states according to Figure 32. Then

$$x_1 = 1 \qquad x_2 = 2$$

$$x_3 = 1 + \frac{1}{2} \cdot x_1 = 1.5 \qquad x_4 = 1 + \frac{1}{3} \cdot x_2 + \frac{2}{3} \cdot x_3 = \frac{8}{3}$$

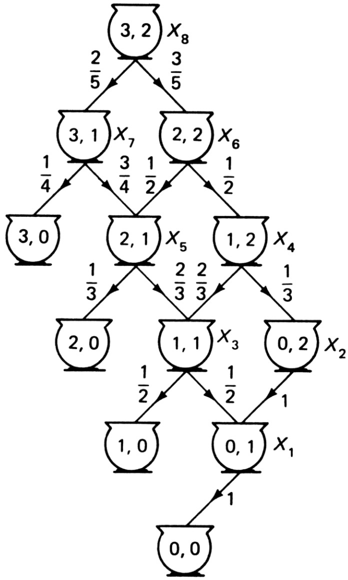


Figure 32

$$x_5 = 1 + \frac{2}{3} \cdot x_3 = 2 \qquad x_6 = 1 + \frac{1}{2} \cdot x_4 + \frac{1}{2} \cdot x_5 = \frac{10}{3}$$

$$x_7 = 1 + \frac{3}{4} \cdot x_5 = \frac{5}{2} \qquad x_8 = 1 + \frac{2}{5} \cdot x_7 + \frac{3}{5} \cdot x_6 = 4$$

On the average, 4 marbles must be taken from the urn to secure all the white marbles. We can, in general, prove that the expectation is $n(m + n + 1)/(n + 1)$. See S. S. Wilks, *Mathematical Statistics*, (New York 1962), p. 143. Wilks calls this problem "the hypergeometric waiting time problem."

6. BUILDING AND DESTROYING TOWERS

It is traditional in probability theory to formulate problems that deal with drawing marbles out of one or several urns. We can also interpret many problems as dealing with the building or destruction of towers.

Exercise 55

This problem is similar to the problem known as "Banach's match boxes." A certain mathematician always carries one match box in his right pocket and one in the left. When he wants a match he selects a pocket at random. Assume the boxes each have N matches at the start. Find the probability distribution of the number of matches in the other box when the mathematician discovers for the first time that a box is empty.

In Lwów, Poland, there was a famous group of mathematicians led by Stefan Banach (1892–1945). This problem, which has been treated extensively in the literature, was originally formulated by H. Steinhaus, another prominent member of the Lwów school, as a joke about Banach's smoking habits. The problem is discussed in W. Feller, *An Introduction to Probability Theory and Its Applications*, 3rd ed., (New York 1968), p. 166.

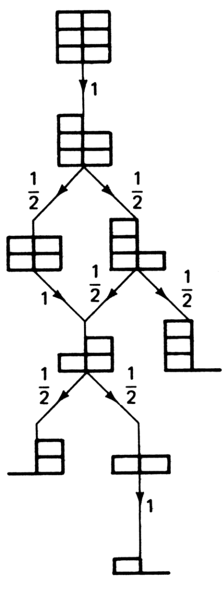


Figure 33

The diagram in Figure 33 illustrates the case for which $m = n = 3$. Let p_k be the probability that altogether k blocks are taken from the towers. It is then seen that

$$p_3 = 1/4, \quad p_4 = 3/8, \quad \text{and} \quad p_5 = 3/8$$

Exercise 56

It follows from the commentary to the preceding exercise that in the case where $m = n = 3$, the expectation is given by

$$\frac{1}{4} \cdot 3 + \frac{3}{8} \cdot 4 + \frac{3}{8} \cdot 5 = 4.125$$

We can derive an expression for the expectation in the general case. (See the reference in the commentary to Exercise 55.)

Exercise 57

The diagram in Figure 34 illustrates the case for which $m = 5$ and $n = 3$. At each possible state the diagram shows the probability that this state is passed during the process of destroying the towers.

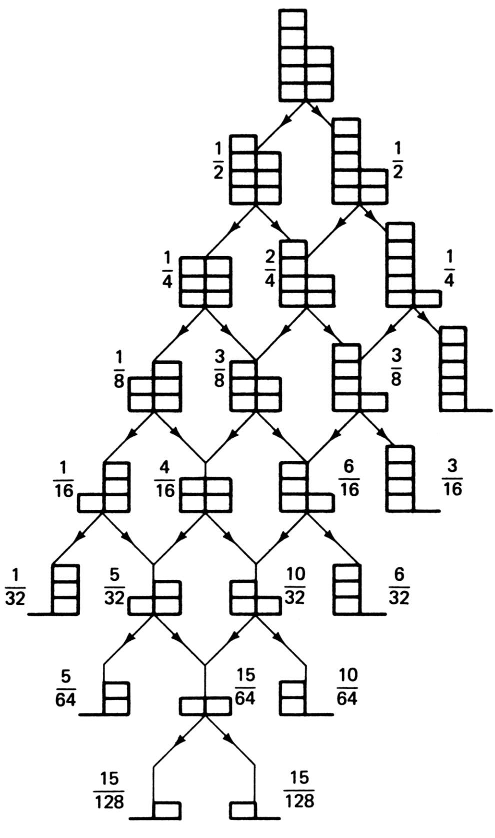


Figure 34

From the diagram we see that the probability of the smaller tower being destroyed in this case is

$$\frac{1}{8}+\frac{3}{16}+\frac{6}{32}+\frac{10}{64}+\frac{15}{128}=\frac{99}{128}\approx 0.773$$

Let q_n be the probability that it is the smaller tower that is finally destroyed, in the case in which originally the larger tower has 10 blocks and the smaller tower has n blocks, where $1 \leqslant n \leqslant 9$. The author obtained the following estimates $\hat{q}_n$ of q_n by simulation. Each estimate was obtained from 100 trials.

$\hat{q}_1 = 1.00$	$\hat{q}_4 = 0.93$	$\hat{q}_7 = 0.80$
$\hat{q}_2 = 0.99$	$\hat{q}_5 = 0.93$	$\hat{q}_8 = 0.73$
$\hat{q}_3 = 1.00$	$\hat{q}_6 = 0.84$	$\hat{q}_9 = 0.60$

Exercises 58–59

The following diagram illustrates the case in which $m = n = 3$.

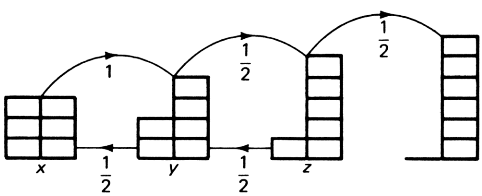


Figure 35

Let x, y, and z be the expected number of blocks moved from the different states according to Figure 35. Then we obtain by conditioning

$$x = 1 + y$$
$$y = 1 + 0.5x + 0.5z$$
$$z = 1 + 0.5y$$

These equations yield $x = 9$. Thus on the average we have to move 9 blocks to get them all in one tower.

Exercise 60

The diagram in Figure 36 illustrates the case in which $m = 5$ and $n = 3$.

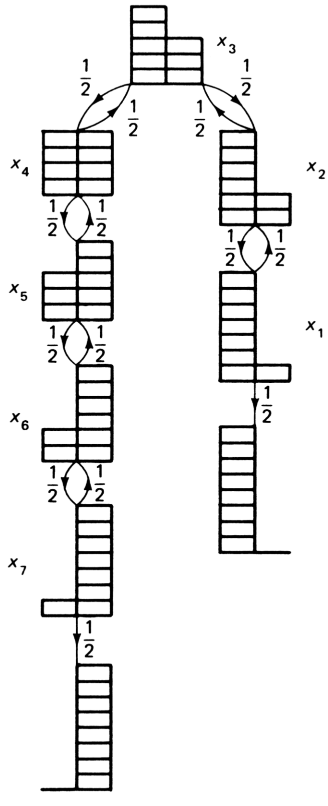


Figure 36

Let x_i be the probabilities that the smaller tower is destroyed from the different states shown in Figure 36. We then obtain

$$x_3 = 0.5x_2 + 0.5x_4 \qquad x_4 = 0.5x_3 + 0.5x_5$$
$$x_2 = 0.5x_3 + 0.5x_1 \qquad x_5 = 0.5x_4 + 0.5x_6$$
$$x_1 = 0.5x_2 + 0.5 \qquad x_6 = 0.5x_5 + 0.5x_7$$
$$x_7 = 0.5x_6$$

These equations yield $x_3 = 5/8$. Thus in this case the probability is 5/8 that the smaller tower is destroyed last.

Exercises 61–62

The diagram in Figure 37 illustrates the case where $m = n = r = 3$. By conditioning, a system of equations can be derived in the usual way to calculate the expected number of blocks taken from the towers. We find it is $409/81 = 5.05$.

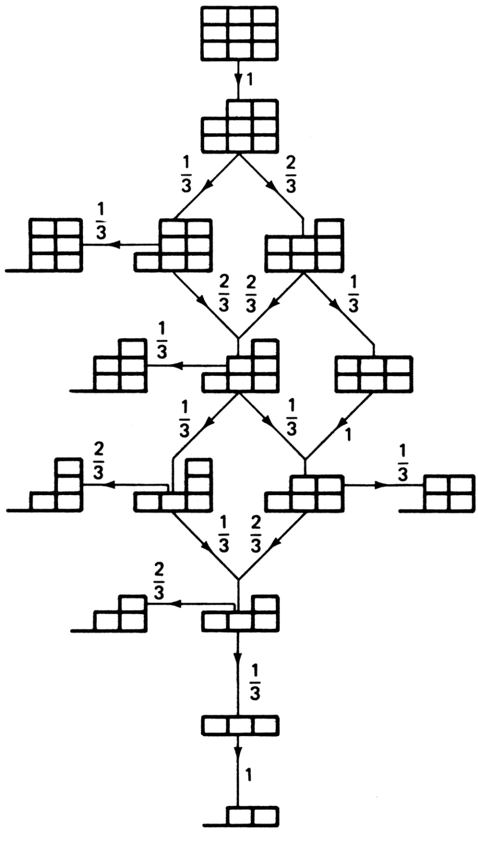


Figure 37

Exercise 63

It is clear that the two variables are negatively correlated. If a large number of blocks have been taken from the towers there should be a small number left in the first tower. If, on the other hand, a small number of blocks have been taken, most of the blocks should be left. The diagram in Figure 38 illustrates an example of results obtained by simulation. The open circles denote two observations. It is an interesting exercise to calculate the correlation coefficient and to fit a straight line to data like those in Figure 38 by the least squares method. Programs for such standard calculations are usually given in the handbook of programmable calculators.

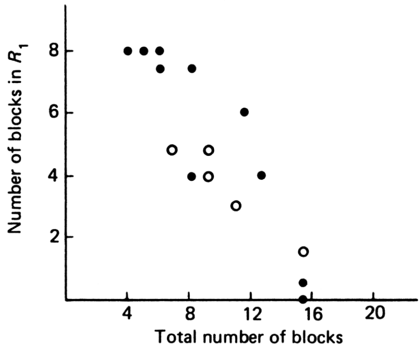


Figure 38

Exercise 64

It is not an easy task to program this calculation in only a few steps. One possibility is to spin a spinner with outcomes 0, 1, 2, 3, 4, and 5 and to let the outcome determine where to jump into the following sequence of program steps. Here "STO +1," for example, means that 1 is added to register 1, and "STO −3" means that 1 is subtracted from register 3.

STO +1	Start here gives "STO +3" and "STO −2"
STO −2	
STO +3	Start here gives "STO +3" and "STO −1"
STO −2	
STO +3	Start here gives "STO +2" and "STO −1"
STO −1	
STO −1	Start here gives "STO +2" and "STO −3"
STO +2	
STO −3	Start here gives "STO +1" and "STO −3"
STO +2	
STO +1	Start here gives "STO +1" and "STO −2"
STO −2	

Exercise 65–66

The diagram in Figure 39 illustrates the case where $m = n = r = 2$.

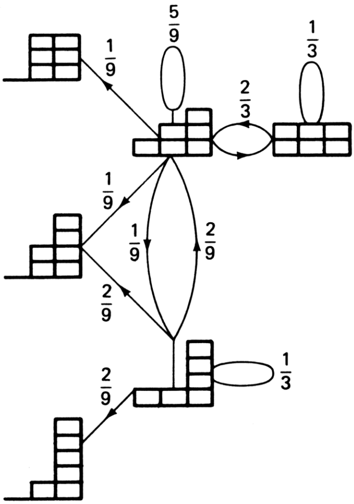


Figure 39

By conditioning in the usual way we can prove that in this case the expected number of moves until one tower is empty is 6.

Exercise 67

This is equivalent to the situation wherein a set consisting of elements of two kinds is sampled at random and without replacement. Such problems have been treated in Exercises 49–54.

7. TOWER GAMES

Exercises 68–69

This game, a very interesting one, has three random characteristics. Which of the players will win? To what heights have the towers increased at the end of a play? What is the duration (expressed in total number of blocks) of a play?

The structure of the game is seen from the diagram in Figure 40. A situation in the game can always be represented by an ordered triple (x, y, z), where x and y represent the colors of the uppermost blocks on the towers, and z denotes which player is to take the next turn in the game. In the figure the colors are represented by 1 and 2; 0 means that so far no block has been placed on one of the towers.

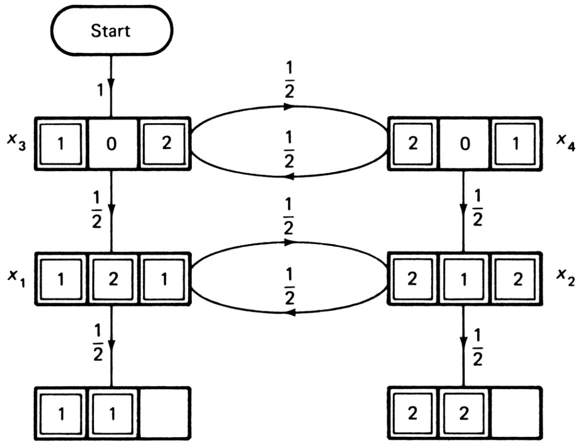


Figure 40

Let x_1, x_2, x_3, and x_4 be the probabilities that Player 1 wins from the different situations in the game as shown in Figure 40. We obtain the following equations:

$$x_1 = 0.5 + 0.5x_2 \qquad x_2 = 0.5x_1$$

$$x_3 = 0.5x_1 + 0.5x_4 \qquad x_4 = 0.5x_3 + 0.5x_2$$

x_3 is the probability that Player 1 wins the game. This probability is found from the equation to be $5/9 \approx 0.56$. Thus the first player has a small advantage in this game.

Instead let x_i be the expected duration of the game from the different situations in the game according to Figure 40. It follows from symmetry that $x_1 = x_2$, and $x_3 = x_4$. Then conditioning gives

$$x_1 = 1 + 0.5x_1$$
$$x_3 = 1 + 0.5x_1 + 0.5x_3$$

The expected duration of the game is $1 + x_3$, which from the equations above is found to be 5.

This game is mentioned in the book *Zufall oder Strategie* by A. Engel, T. Varga, and W. Walser (Klett Verlag, Stuttgart 1974). In this book some variations of the game are also described. These and some further variations of the game are treated in the exercises that follow.

We can mention finally that the probability of the duration of the game being n moves can be shown to be $(n-2)/2^{n-1}$, $n > 2$.

Exercises 70–72

The structure of this game is seen from Figure 41. A situation in this game is described by four numbers, where the first three numbers give the colors on the uppermost blocks on the towers, and the fourth number indicates which player is to make the next move.

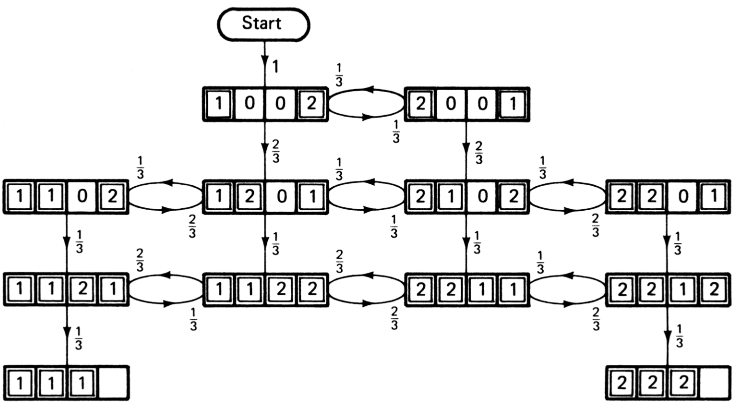


Figure 41

With the aid of the diagram in Figure 41 you can calculate the winning probabilities for the two players and also the expected duration of the game, but it is a tedious task. We find that the first player wins with probability 0.508, so that in this game the first player is favored. The expected duration is found to be 16.75.

Exercise 73

The book mentioned in the commentary of Exercises 68–69 gives 53/105 = 0.505 as the winning probability for Player 1 in this case. The expected duration is 52.73, so it takes quite a long time to finish a game.

Exercises 74–76

The structure of this game is seen from the diagram in Figure 42. We find after lengthy calculation that Player 1 wins with probability 81/217 = 0.37, that Player 2 wins with probability 72/217 = 0.33, and that Player 3 wins with probability 64/217 = 0.29. We have not calculated the expected duration but simulations indicate that it is approximately 9 moves.

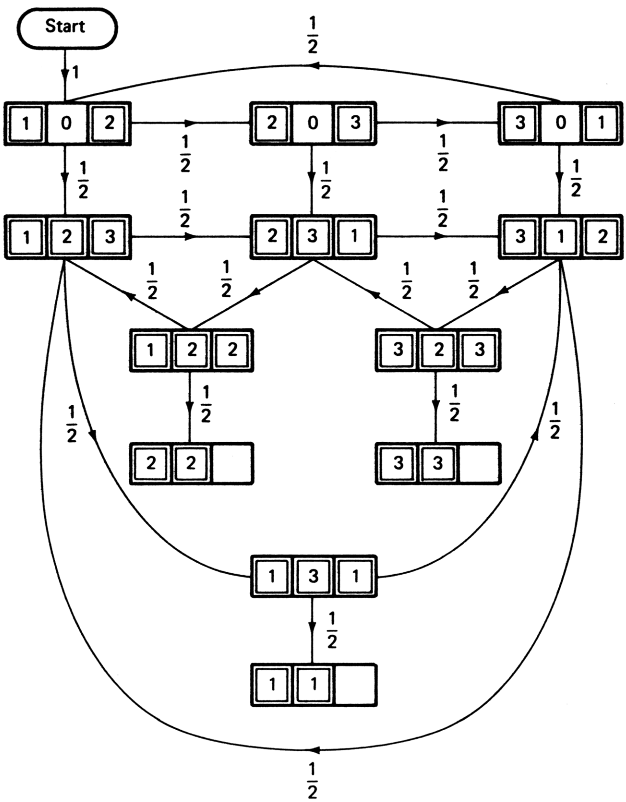


Figure 42

Exercise 77

It is possible to write a program so that by changing one step in the program we change the number of players. It is laborious to execute these games with more than 3 players by analytical methods, so it is more convenient to study them by simulation. With 4 players the author obtained from 100 plays an average duration of 16.91 moves, and with 5 players an average duration of 33.84 moves. It takes considerable time to make such simulations on a programmable calculator.

Exercise 78

The reader is challenged to draw an arrow diagram which describes this game. The plays of the game usually take a very long time. The following table gives the results of 10 simulations the author has made.

Simulation number	1	2	3	4	5	6	7	8	9	10
Number of moves	83	128	46	89	35	101	91	61	51	113
Winning player, in order	2	2	1	2	2	2	1	1	3	2

Exercise 79

This game is simple to play and to simulate but has a complicated structure, as you see from the arrow diagram in Figure 43, which describes the game. The numbers in the upper right corners indidate which player is to make the next move. The heavy-framed boxes indicate the winning positions. (This complex diagram was devised by the author's 11-year-old son, Johan.)

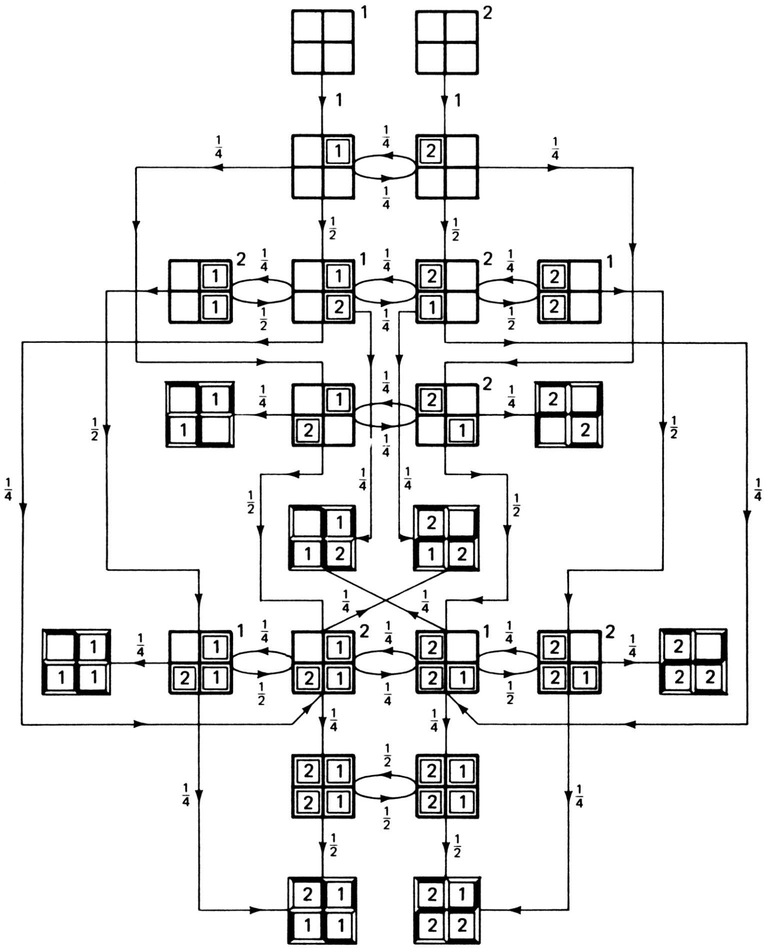


Figure 43

8. RUNS AND OTHER PATTERNS

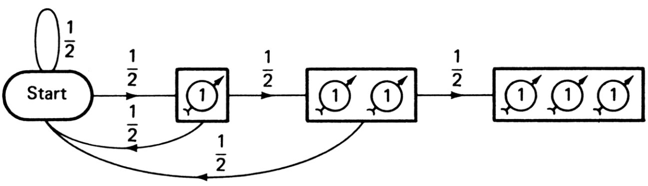


Figure 44

Exercises 80–81

The diagram in Figure 44 illustrates the case where $k = 3$. Let the expected duration in this case be x. With aid of the diagram we obtain the equation

$$x = \frac{1}{2} \cdot (1 + x) + \frac{1}{4} \cdot (2 + x) + \frac{1}{8} \cdot (3 + x) + \frac{1}{8} \cdot 3$$

This equation gives $x = 14$. Thus, on the average, you have to spin the spinner 14 times to get three successive 1s.

A similar argument shows that the expected number of spins to produce k successive 1s is $2^{k+1} - 2$.

Exercises 82–83

Let μ be the expectation and σ^2 the variance of the waiting time. With $q = 1 - p$ the following formulas hold:

$$\mu = \frac{1 - p^k}{qp^k} \quad \text{and} \quad \sigma^2 = \frac{1}{(qp^k)^2} - \frac{2k + 1}{qp^k} - \frac{p}{q^2}$$

The expectation can be found as in Exercises 80 and 81. The variance is most easily found with the aid of a probability generating function, a useful tool in probability theory. See W. Feller, *An Introduction to Probability Theory and Its Applications*, 3rd ed., (New York 1968), p. 324.

Exercises 84–85

Observe that in this case the pattern desired consists of k successive equal outcomes with no conditions as to what the outcome should be. We can prove tnat the expected waiting time is

$$\frac{N^k - 1}{N - 1}$$

For $n = 10$ and $k = 3$, for instance, the expected waiting time is 111. Thus we must generate on the average 111 random digits to get three successive equal digits. See L. Råde, *Waiting for Patterns in a Sequence of Random Numbers*, Zeitschrift für Angewandte Mathematik und Mechanik 56, 1976.

Exercises 86–89

In programming these problems, keep in mind that the number of runs is one more than the number of times two successive elements in the sequence are different. Each time two successive elements are different, a new run starts, and the first element also starts a run. This fact can also be used to prove that the expected number of runs, when the spinner in Figure 17 is spun n times, is

$$1 + 2(n-1)p(1-p)$$

Exercises 90–91

We can prove that the expected waiting time in this case is

$$\left(\frac{N}{N-1}\right)^n$$

and that the variance is given by

$$\left(\frac{N}{N-1}\right)^N \left[\frac{3N-1}{N-1} - \left(\frac{N}{N-1}\right)^N\right]$$

This is shown in the reference mentioned in the commentary to Exercises 84 and 85. Observe that for a large N the expectation is close to e, and the variance is close to $e(3-e)$.

Exercise 92

A general formula for the expected waiting time for this pattern is unknown to the author. The special case where $n = k = 3$ is portrayed by the arrow diagram in Figure 45. In this case the pattern to be generated is 0 1 2.

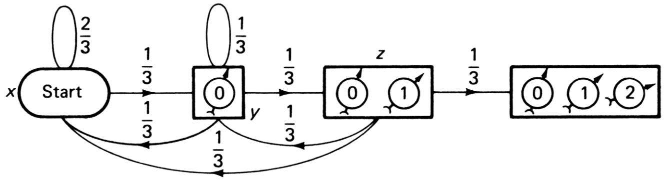


Figure 45

If x, y, and z are the expected waiting times from the states as shown in Figure 45, we obtain

$$x = 1 + \frac{2}{3} \cdot x + \frac{1}{3} \cdot y$$

$$y = 1 + \frac{1}{3} \cdot y + \frac{1}{3} \cdot x + \frac{1}{3} \cdot z$$

$$z = 1 + \frac{1}{3} \cdot y + \frac{1}{3} \cdot x$$

These equations give $x = 27$; that is, in this case on the average 27 spins have to be made to generate the pattern 0 1 2.

Exercise 93

According to the reference mentioned in the commentary to Exercises 84 and 85, the expected waiting time in this case is

$$\frac{(N + 1)^2}{N + (-N)^{-N}}$$

Exercise 94

A general formula for the expectation is not known, but can be derived with the same kind of reasoning used to derive the expectation in Exercise 93.

Exercise 95

For this problem, see W. Feller, *An Introduction to Probability Theory and Its Applications*, 3rd ed., (New York 1968), p. 278.

Exercises 96–97

When programming here, keep in mind that you obtained a palindrome when for the first time an outcome is equal to the outcome obtained on the first spin. You can prove that the expectation is 3, independent of what value p has. It is also interesting to study how the variance of the waiting time depends upon p.

9. RANDOM WALKS

Random walks have been studied extensively; results of the studies fill volumes of literature on the subject. They are also usually treated in elementary textbooks in probability. Random walks have important applications, but in addition are also very interesting from a theoretical point of view.

Exercises 98–99

This is a classical problem in probability theory with a very long history. The random walk can be interpreted as a fair game between two players who at the start have capitals consisting of a and b dollars (or other monetary units). In each play the losing player gives one dollar to the winning player. The position of the particle can then be interpreted as the total gain of that player who from the beginning had a capital of a dollar. At absorption in b this player has defeated the other player and at absorption in $-a$ he is himself defeated.

We can prove that absorption will take place in the point $-a$ with probability $b/(a + b)$ and in the point b with probability $a/(a + b)$. The sum of these two probabilities is 1. Absorption thus takes place in one of the points with probability 1; that is, the probability is 0 for the event that the particle will go on jumping forever.

We can also prove that the expected number of steps until absorption is given by the product ab, that is, the product of the two players' initial capital. This implies that random walks, for example, between -4 and 4 and between -1 and 16 have the same expected duration. However, even if the expectations are the same, simulations will show that the random variation of the durations in these cases differ widely.

When you simulate these random walks you might find it useful to use the spinner in Exercise 38. Compare also with Exercise 60.

Exercises 100–101

We can prove that the particle returns to the origin with probability 1, but that the expected number of steps until this return is infinite. Simulations of these random walks now and then give very long walks, and the mean of the walks does not stabilize around a fixed value. How many of the walks can be expected to be finished after one step?

Exercise 102

In this case there are not such simple formulas for the expected duration and for absorption probabilities as in the symmetric case treated in Exercises 100 and 101. We can prove that the expected duration is $(q = 1 - p)$

$$\frac{a}{q-p} - \frac{a+b}{q-p} \cdot \frac{1-(q/p)^a}{1-(q/p)^{a+b}}$$

and that the probability for final absorption in $-a$ is

$$\frac{(q/p)^{a+b} - (q/p)^a}{(q/p)^{a+b} - 1}$$

On the other hand, the programming of these random walks is no more difficult than the symmetric ones.

Exercises 103–104

We can prove that the expected number of returns to the origin in $2n$ steps is approximately

$$2\sqrt{n/\pi} - 1$$

If the particle makes, for example, ten jumps ($n = 5$), the expected number of returns is approximately 1.52.

Exercise 105

One possible way to simulate these random walks is to give the corners of the cube coordinates as shown in Figure 46. The random walk can then be simulated by successfully choosing one of the coordinates at random and multiplying the chosen coordinate by -1.

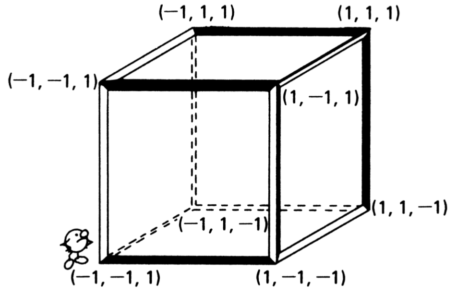


Figure 46

Another possibility is to use the description given by the arrow diagram in Figure 47. Here the numbers of the states represent the distances from the actual position of the particle from the origin. The distance is calculated as the shortest distance along the edges of the cube.

We can prove that the expected duration is 10 steps.

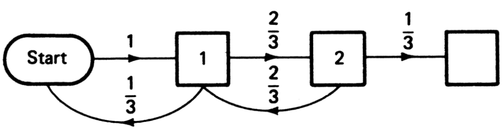


Figure 47

Exercise 107

Figure 48 shows a four dimensional cube. The coordinates of the starting and finishing points of the walk are shown in the figure.

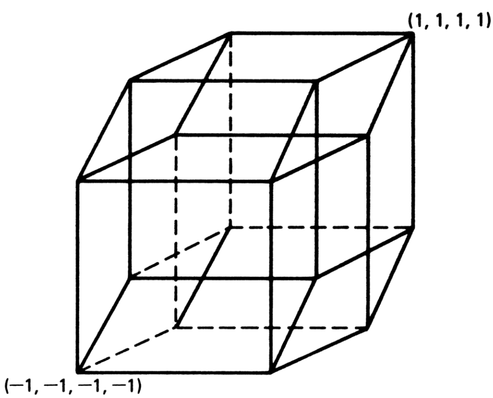


Figure 48

Exercise 109

These random walks are described by the arrow diagram in Figure 49. An open circle in a corner of the square shows that this point has already been visited. The filled circle shows the actual position of the particle.

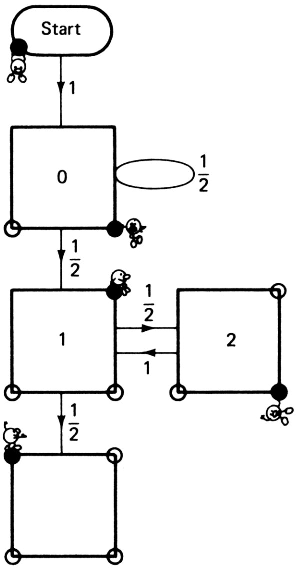


Figure 49

We can prove that the expected duration of this random walk is 6. For the corresponding random walk on a polygon with n corners the expected duration is $\binom{n}{2}$. See L. Råde, "Random Walks on the Hexagon," *International Journal on Mathematical Education in Science and Technology*, vol. 6, 1976, pp. 255–263.

10. SOME STATISTICAL APPLICATIONS

Exercise 112

You can make your calculator generate a probability that is unknown to you in the following way. Generate random numbers

$x_0, x_1, x_2, \ldots$ in the same way as with the 147-generator, and use x_9, for example, as probability p.

Exercise 113

You can make your calculator generate a number N that is unknown to you. Generate $x_0, x_1, x_2, \ldots$ as in Exercise 112 and use $1000x_9$, for example, as the value for N.

We see by the following argument that $2\overline{x}$ is a suitable estimate of N. The mean $\overline{x}$ is a suitable *estimate* of the midpoint of the distribution and thus $2\overline{x}$ should be a suitable *estimate* of the right endpoint of the distribution. More precisely, the mean $\overline{x}$ is an unbiased estimate of the expectation in this experiment as it is in every random experiment. The expectation is, in this case,

$$\frac{1}{N} \cdot 1 + \frac{1}{N} \cdot 2 + \ldots + \frac{1}{N} \cdot N = \frac{1+N}{2}$$

Thus $\overline{x}$ is a suitable estimate of $(1+N)/2$, which implies that $2\overline{x} - 1$ is a suitable estimate of N. You should compare the accuracy of the two estimates $2\overline{x}$ and $2\overline{x} - 1$ by simulation.

An example of a practical situation dealing with this estimation process follows. A certain group of objects are numbered 1, 2, . . . , N. The numbers of some such randomly chosen objects are observed and this information is used to estimate the total number N of objects. For instance, a certain type of sail boat is observed, and from the number of sails we try to estimate the total number of such boats. In this case, however, it would be safer to simply ask the manufacturer of the sail boats how many they produced. It is said that during World War II such methods were used to estimate enemy production, in which case it was impossible to ask the manufacturer for the value of N.

Exercise 114

We use similar reasoning to that given in the preceding commentary to understand why $\frac{n+1}{n} x_{\max}$ is a suitable estimate of N. On the average, the n observations should be uniformly distributed over the interval from 1 to N; that is, if the observations are ordered according to size, the distances between successive observations and also the distances between the endpoints and the smallest and largest observations should be, on the average, the same. Then the entire interval (draw a figure, for instance, for $n = 4$) from 1 to N consists of $(n + 1)$ such intervals, of which n have the total length x_{max}. The given argument is not precise but more precise reasoning gives nearly the same result.

Exercise 115

An estimate of an unknown parameter is called consistent if it has the following attractive property. When the number of observations increases, the estimate converges (in a way we do not describe here) to the unknown parameter. We can prove that the relative frequency is a consistent estimate of the corresponding probability. As a matter of fact, this is a famous theorem in probability, usually called *the Bernoulli theorem*, which in turn is a special case of another theorem, the *law of large numbers*. It is said that Jakob Bernoulli (1654–1705) needed 15 years to prove his theorem.

Exercise 116

We can prove that $2\overline{x} - 1$ and $\frac{n+1}{n}x_{max}$ are consistent estimates of N.

Exercise 118

The two estimates considered here are both unbiased estimates of N. One way to compare them is to consider their random variation. The one with smaller random variation is, of course, preferable. In statistics an estimate usually is described as more *effective* than another estimate if it has a smaller random variation. This variation can be measured by the variance of the estimate, which usually can be calculated according to a formula. Or it can be estimated by simulation as in this exercise.

Your simulations should indicate that $\frac{n+1}{n}x_{max}$ is more efficient than $2\overline{x} - 1$. Observe that $2\overline{x} - 1$ and $2\overline{x}$ have the same random variation. The subtraction of 1 does not change the random variation.

Exercise 119

A random interval that contains an unknown parameter (for example, the probability p) with a known probability is called (in statistics) a *confidence interval.* The probability that the interval contains the parameter is called its *confidence level.* We can prove that

$$p = \hat{p} \pm 2\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

is a confidence interval for p with confidence level ≈ 0.95. The confidence level 0.95 is used often. More precisely, the factor 1.96 should be used instead of 2, but the difference is of no significance.

11. MISCELLANEOUS PROBLEMS

Exercise 121

The following simple argument shows that the expected number of children within the limits described in Exercise 121 will be 3. After the first child is born, a family must bear on the average two more children to have a child of a sex different from the first child. In the general case when the probability for a boy is p and the probability for a girl is q, $p + q = 1$, the probabilities for different family sizes and the expected number of children can be obtained from the diagram in Figure 50.

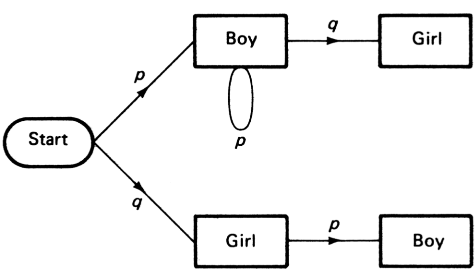


Figure 50

The diagram shows that the expected number of children is

$$1 + p \cdot \frac{1}{q} + q \cdot \frac{1}{p}$$

which, after some algebraic manipulation, can be written

$$\frac{1 - pq}{pq}$$

For $p = 0.52$ and $q = 0.48$, for instance, this expectation is 3.0064. The families in Exercise 121 will have more k children with probability $p^k + q^k$, because p^k is the probability for k successive boys and q^k is the probability for k successive girls. This can also be taken as the starting point of a mathematical analysis of the stopping rule in Exercise 121.

Concerning the stopping rule in Exercise 122, see the commentary to Exercise 34. To study the rule in Exercise 123, the diagram in Figure 51 can be used.

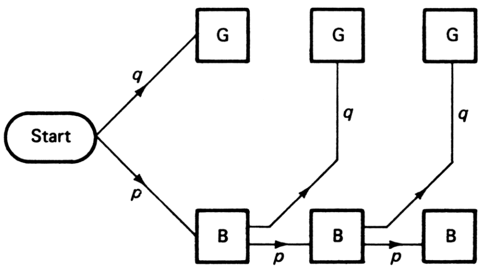


Figure 51

None of the stopping rules in these exercises will change the sex ratio in the general population. This is because it is not possible by using stopping rules to alter the frequencies of outcomes in a random game. For a mathematical proof, see the book A. Engel, *Wahrsscheinlichkeitsrechnung und Statistik*, Volume 1 (Klett Verlag 1973), p. 137. In his book, Engels mentions that the sociologist S. Winston suggested that such stopping rules might explain why more boys than girls are born. It goes without saying that he was incorrect!

Finally, we can mention that the stopping rule in Exercise 121 is a special case of the following situation. A certain random experiment has N equally probable outcomes. Trials are made until all outcomes have occurred. We can prove that the expected number of trials to obtain all the different outcomes is

$$N\left(1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{N}\right)$$

Calculate this expectation on your calculator for the case $N = 10$ (of interest if you start to collect random digits) and $N = 365$ (of interest if you start making friends so that you can go to a birthday party each day of the year).

Exercises 126–127

This random experiment is described by the arrow diagram in Figure 52. We leave it to the industrious reader to find an expression for the expectation and to compare the theoretical value with that found by simulation. We can reveal that in the case where $p = 0.5$, the expectation is 10.

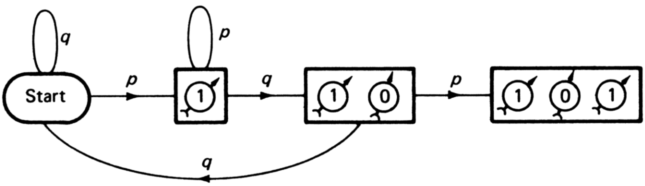


Figure 52

Exercises 128–130

Let $P(N, n)$ be the probability that the nth spin of the spinner gives a record. It then follows that for $n \geqslant 2$

$$P(N, n) = \frac{1^{n-1} + 2^{n-1} + 3^{n-1} + \ldots + (N-1)^{n-1}}{N^n}$$

This follows by considering the different possibilities of the nth spin of the spinner. Suppose, for example, that the nth spin has given 3. If 3 is to be a record, the preceding $n - 1$ spins all must have given numbers smaller than 3; that is, 0, 1, or 2. The number of possibilities for this occurrence is 3^{n-1}.

The probability $P(N, n)$ can be written

$$P(N, n) = \frac{1}{N}\left[\left(\frac{1}{N}\right)^{n-1} + \left(\frac{2}{N}\right)^{n-1} + \left(\frac{3}{N}\right)^{n-1} + \ldots + \left(\frac{N-1}{N}\right)^{n-1}\right]$$

But this expression implies that $P(N, n)$ is the total area of the rectangles in Figure 53, which in turn implies that for large N

$$P(N, n) \approx \int_0^1 x^{n-1}\, dx = \frac{1}{n}$$

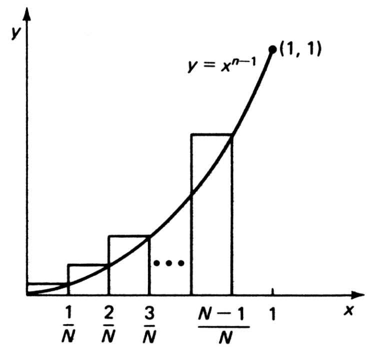


Figure 53

Thus for large N the probability is $1/n$ that the nth spin will produce a record, which implies that for large N the expected number of records in n spins is approximately

$$1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n}$$

Exercises 131–133

The birthday problem is a classic problem in probability and is found in most elementary textbooks on the subject. The fact that

$P(365, n)$ is larger than 0.5 already for $n = 23$ astonishes many and can be used to advantage when betting. It may be that the astonishment comes from confusing the situations in Exercises 132 and 133. The author was once present at a meeting of about 30 mathematicians where we started the day by honoring a German professor whose birthday was that day. Then the chairman, who obviously was not a probabilist, announced that according to what he had read there must be one person more with a birthday that day. That remark referred to the situation in Exercise 133 where the probability p_n is given by

$$p_n = 1 - \frac{364}{365}^{\,n}$$

Here we have to give n a rather high value to make this probability more than 0.5.

Exercises 134–137

It can be proved that the probability of there always being more votes for A than for B throughout the counting is simply given by

$$\frac{a-b}{a+b}$$

This remarkable result was first proved by W. A. Whitworth in 1878 and independently by J. Bertrand in 1887. Much research has been conducted around this problem; the research has been found to have unexpected applications, for example, in queuing theory, which is discussed in Exercises 138–140. The problem is treated in W. Feller, *An Introduction to Probability Theory and Its Applications*, vol. 1, 3rd ed., (New York 1968), p. 69.

Exercises 138–140

Queuing theory, or the theory of stochastic service systems, is an important area of applied probability theory. There exists a vast amount of literature on the subject. The basic principles are often treated in elementary textbooks on probability.

These exercises treat a discrete version of what is usually (with a standardized kind of notation) called the M/M/1 queuing system.

When programming Exercise 138 you might find it useful to use the flow chart in Figure 54. Here the discrete time epochs are registered in register R_1 and the number of customers in the system is stored in register R_2.

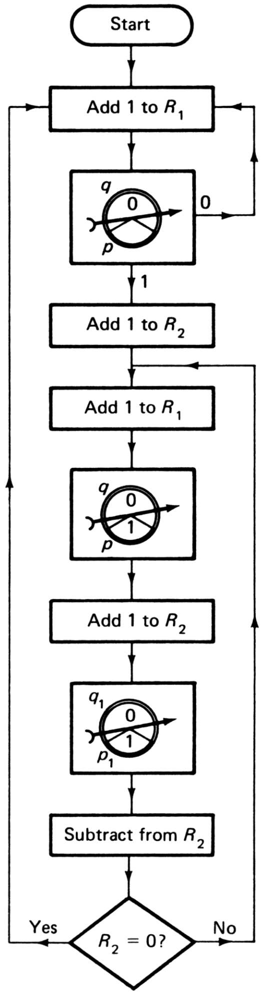


Figure 54

Exercises 141–143

We can show that $(r - 1)/(N - 1)$ is an unbiased estimate of the probability p but not so r/N.

APPENDIX 1. A WORKED EXAMPLE

A symmetric coin with outcomes 0 and 1 is tossed until for the first time three successive tosses all give the outcome 1 *or* two successive tosses all give the outcome 0. In other words, the sequence of tosses is to be counted until the pattern 1 1 1 or the pattern 0 0 is obtained. Let us analyze this random experiment. It is described by the arrow diagram in Figure 55.

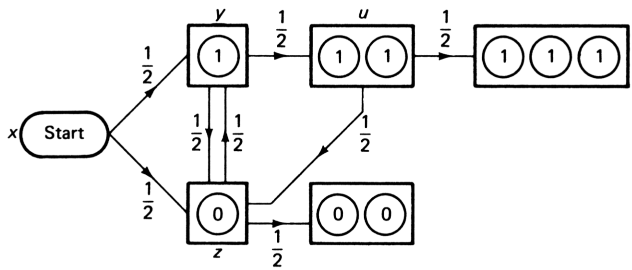


Figure 55

Let x be the probability that the sequence of tosses ends with the pattern 0 0. Let y be the corresponding probability after the first toss has given 1, let z be the corresponding probability after the first toss has given 0, and finally let u be the corresponding probability after the first two tosses both have given 1. These so-far-unknown probabilities have been written down at the proper positions in the diagram in Figure 55.

Then we obtain

$$x = \frac{1}{2} \cdot y + \frac{1}{2} \cdot z$$

Here $\frac{1}{2} \cdot y$ is the probability that the first toss gives 1 and that after that the sequence ends with 0 0, and $\frac{1}{2} \cdot z$ is the probability that the first toss gives 0 and that the sequence ends with 0 0. Similar arguments give

$$y = \frac{1}{2} \cdot z + \frac{1}{2} \cdot u$$

$$z = \frac{1}{2} \cdot y + \frac{1}{2} \cdot 1$$

$$u = \frac{1}{2} \cdot z + \frac{1}{2} \cdot 0$$

These equations give $x = 0.7$. Thus, for instance, in 100 trials of this random experiment it is to be expected that about 70 trials will end with the pattern 0 0 and about 30 with the pattern 1 1 1.

Another possible way to find the probability x is to start with the equation

$$x = \frac{1}{2} \cdot y + \frac{1}{2} \cdot z$$

and then write down equations that only contain y and z, respectively. The equation for y is

$$y = \frac{1}{4} \cdot y + \frac{1}{8} \cdot y + \frac{1}{4} + \frac{1}{8}$$

The equation can be derived. Look at the situation when the first toss has given 1; see Figure 55. Then $\frac{1}{4} \cdot y$ is the probability that the next two tosses give 0 1 and that after that the sequence ends with 0 0. Observe that 0 1 takes the sequence of tosses back to where it started. Furthermore, $\frac{1}{8} \cdot y$ is the probability that the next three tosses give 1 0 1 and that after that the pattern 0 0 "wins." Finally, $\frac{1}{4}$ is the probability that the next two tosses give 0 0, and $\frac{1}{8}$ is the probability that the next three tosses give 1 0 0.

A similar argument yields

$$z = \frac{1}{4} \cdot z + \frac{1}{8} \cdot z + \frac{1}{2}$$

These equations give $x = 0.7$ as before.

Now let x be the expected number of tosses necessary to produce one of the patterns 0 0 and 1 1 1. Let y be the expected number of tosses, given that the first toss has produced the outcome 1. Here y does not include the first toss. Let z and u have the corresponding meaning, given that the first toss has produced 0 or that the first two tosses have yielded 1 1; see Figure 55. Then

$$x = \frac{1}{2} \cdot (1 + y) + \frac{1}{2} \cdot (1 + z)$$

This expression is derived as follows. With probability $\frac{1}{2}$, the first toss will produce 1 and in this case the expected number of tosses is $(1 + y)$; with probability $\frac{1}{2}$ the first toss will yield 0 and in this case the expected number of tosses is $1 + z$.

Similar arguments give

$$y = \frac{1}{2} \cdot (1 + z) + \frac{1}{2} \cdot (1 + u)$$

$$z = \frac{1}{2} \cdot (1 + y) + \frac{1}{2} \cdot 1$$

$$u = \frac{1}{2} \cdot (1 + z) + \frac{1}{2} \cdot 1$$

These equations give $x = 4.2$. Thus on the average 4.2 tosses are needed to produce the pattern 0 0 or the pattern 1 1 1.

Another (perhaps better) method to find the expectation x is described here. Start as above with an equation for x:

$$x = \frac{1}{2} \cdot (1 + y) + \frac{1}{2} \cdot (1 + z)$$

Then write down equations that only contain y and z, respectively. The equation for y is

$$y = \frac{1}{4} \cdot (2 + y) + \frac{1}{8} \cdot (3 + y) + \frac{1}{4} \cdot 2 + \frac{1}{4} \cdot 2 + \frac{1}{8} \cdot 3$$

The term $\frac{1}{4} \cdot (2 + y)$ can be explained as follows. The factor $\frac{1}{4}$

is the probability that the first two tosses (after the very first toss has given 1) yield 0 1 (see Figure 55); in this case the expected number of tosses is $2 + y$. The other terms are obtained in a similar way. The equation gives $y = 3.6$. Similar arguments give the following equation for z:

$$z = \frac{1}{4} \cdot (2 + z) + \frac{1}{8} \cdot (3 + z) + \frac{1}{2} \cdot 1 + \frac{1}{8} \cdot 3$$

This random experiment can be simulated on a programmable calculator. Thus 100 simulated trials resulted in 73 trials ending with the pattern 0 0, which gives the estimate 0.73 of the probability p that 0 0 "wins." A confidence interval (see Exercise 119) for p with the (approximate) confidence level 0.95 is then given by

$$p = 0.73 \pm 2 \sqrt{\frac{0.73 \cdot 0.27}{100}} = 0.73 \pm 0.09$$

Furthermore 1000 simulated trials gave the estimate 695/1000 of p. The corresponding confidence interval is

$$p = 0.695 \pm 2 \sqrt{\frac{0.695 \cdot 0.305}{1000}} = 0.695 \pm 0.029$$

The number of tosses required to produce the pattern 0 0 or the pattern 1 1 1 can also be studied by simulation. You can, for instance, let your calculator generate sequence by sequence. If each time you write down the length of the sequence you can then draw a bar diagram to show the observed variations of the numbers of tosses. Or you can program the calculator to make a number of sequences and to calculate the mean $\overline{x}$ and the standard deviation s of the number of tosses. Thus 100 simulated trials gave as the mean $\overline{x} = 3.96$ and the standard deviation $s = 2.28$. If the expected number of tosses is μ, the confidence interval with confidence level 0.95 is

$$\mu = 3.96 \pm 2 \cdot \frac{2.28}{\sqrt{100}} = 3.96 \pm 0.46$$

Furthermore, 5000 trials gave the mean $\overline{x} = 4.1532$ and the standard deviation $s = 2.3682$. The confidence interval is

$$\mu = 4.1532 \pm 2 \cdot \frac{2.3682}{\sqrt{5000}} = 4.15 \pm 0.07$$

REFERENCES

The following works are mentioned in the text.

A. Engel, *Wahrscheinlichkeitsrechnung und Statistik*, 2 vols., (Stuttgart: Klett Verlag, 1973-76).

A. Engel, T. Varga, and W. Walser, *Zufall oder Strategie?*, (Stuttgart: Klett Verlag, 1974).

W. Feller, *An Introduction to Probability Theory and Its Applications*, vol. 1, 3rd ed., (New York: Wiley, 1968).

G. S. Fishman, *Concepts and Methods in Discrete Event Digital Simulation*, (New York: Wiley-Interscience, 1973).

T. E. Hull and A. R. Dobell, *Random Number Generators*, Siam Review, vol. 4 (1962).

B. Jansson, *Random Number Generators*, (Stockholm, 1966).

L. Råde, *Thinning of Renewal Point Processes*, (Gothenburg: Matematisk Statistik AB, 1972).

L. Råde, "Waiting for Patterns in a Sequence of Random Numbers," *Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 56, (1976).

L. Råde, "Random Walks on the Hexagon," *International Journal on Mathematical Education in Science and Technology*, vol. 6 (1975).

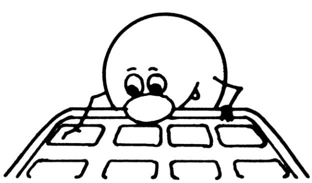
S. S. Wilks, *Mathematical Statistics*, (New York: Wiley, 1962).

Several exercises have been inspired by the highly recommendable books by A. Engel; see the first reference above. The following paper has also been used:

A. Engel, "Computing and Probability," in *Statistics at the School Level*, ed.: L. Råde (Stockholm: Almqvist & Wiksell International, 1975).

part three

PROGRAMS



EXERCISE 2

Before starting store x_0 in R_0. Each time the R/S-key is pressed, four random digits are obtained.

01	RCL 0	07	STO 0
02	1	08	EEX
03	4	09	4
04	7	10	x
05	x	11	f INT
06	g FRAC	12	GTO 00

EXERCISE 3

Change program step 09 in the program in Exercise 2 to "1".

EXERCISE 4

(a) Before starting store x_0 in R_0. In the pause the number of digits which have been generated thus far are shown. When the program stops the number of zeros is displayed. The program can be repeated by pressing the R/S-key.

01	RCL 0	15	STO +1
02	1	16	1
03	4	17	STO +2
04	7	18	RCL 2
05	x	19	f PAUSE
06	g FRAC	20	EEX
07	STO 0	21	n
08	1	22	$f\ x \neq y$
09	0	23	GTO 01
10	x	24	RCL 1
11	f INT	25	R/S
12	$g\ x \neq 0$	26	0
13	GTO 16	27	STO 1
14	1	28	STO 2
		29	GTO 01

(b) In this case the number of fives is displayed when the program stops. Before starting, store x_0 in R_0.

01	RCL 0	05	x
02	1	06	g FRAC
03	4	07	STO 0
04	7	08	1

```
09 0
10 x
11 f INT
12 5
13 –
14 g x≠0
15 GTO 18
16 1
17 STO +1
18 1
19 STO +2
20 RCL 2
21 f PAUSE
22 EEX
23 n
24 f x≠y
25 GTO 01
26 RCL 1
27 R/S
28 0
29 STO 1
30 STO 2
31 GTO 01
```

EXERCISE 5

The following program uses the method described in Figure 26. Before starting store x_0 in R_6 and 147 in R_7. When the program stops, the frequencies for the digits 0–4 can be recalled from the registers $R_0 - R_4$. Before the program is repeated 0 must be stored in $R_0 - R_5$.

```
01 RCL 6
02 RCL 7
03 x
04 g FRAC
05 STO 6
06 1
07 0
08 x
09 f INT
10 g x=0
11 GTO 40
12 1
13 –
14 g x=0
15 GTO 37
16 1
17 –
18 g x=0
19 GTO 34
20 1
21 –
22 g x=0
23 GTO 31
24 1
25 –
26 g x≠0
27 GTO 42
28 1
29 STO +4
30 GTO 43
31 1
32 STO +3
33 GTO 43
34 1
35 STO +2
36 GTO 43
37 1
38 STO +1
39 GTO 43
40 1
41 STO +0
42 1
43 STO +5
44 RCL 5
45 f PAUSE
46 EEX
47 n
48 f x≠y
49 GTO 01
```

The following program uses the method described in Figure 27. Before starting store x_0 in R_6 and 147 in R_7.

```
01 RCL 6
02 RCL 7
03 x
04 g FRAC
05 STO 6
06 1
07 0
08 x
09 f INT
10 3
11 x↔y
12 f x<y
13 GTO 27
14 3
15 f x=y
16 GTO 24
17 ↓
18 4
19 f x≠y
20 GTO 41
21 1
22 STO +4
23 GTO 42
24 1
25 STO +3
26 GTO 42
27 g x=0
28 GTO 38
29 1
30 f x=y
31 GTO 35
32 1
33 STO +2
34 GTO 42
35 1
36 STO +1
37 GTO 42
38 1
39 STO +0
40 GTO 42
41 1
42 STO +5
43 RCL 5
44 f PAUSE
45 EEX
46 n
47 f x≠y
48 GTO 01
49 GTO 00
```

EXERCISE 6

The following program is a shortened version of the second program in Exercise 5. The frequencies for the digits 0–4 are stored in R_0–R_4. If the program steps 11–14 are changed to 11 CHS, 12 ENTER, 13 9, and 14 + the frequencies for digits 9–5 are stored in R_0–R_4. Before starting store x_0 in R_6 and 147 in R_7.

```
01 3
02 RCL 6
03 RCL 7
04 x
05 g FRAC
06 STO 6
07 1
08 0
09 x
10 f INT
11 g NOP
12 g NOP
13 g NOP
14 g NOP
15 f x<y
16 GTO 28
```

```
17 f x=y
18 GTO 25
19 4
20 f x≠y
21 GTO 41
22 1
23 STO +4
24 GTO 42
25 1
26 STO +3
27 GTO 42
28 g x=0
29 GTO 38
30 1
31 f x=y
32 GTO 36
33 1
34 STO +2
35 GTO 42
36 STO +1
37 GTO 42
38 1
39 STO +0
40 GTO 42
41 1
42 STO +5
43 RCL 5
44 f PAUSE
45 EEX
46 n
47 f x≠y
48 GTO 01
49 GTO 00
```

EXERCISE 7

Before starting store x_0 in R_0. When the program stops, the frequency for odd digits is displayed. The program can be repeated.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 1
09 0
10 x
11 f INT
12 2
13 ÷
14 g FRAC
15 g x=0
16 GTO 19
17 1
18 STO +1
19 1
20 STO +2
21 RCL 2
22 f PAUSE
23 EEX
24 n
25 f x≠y
26 GTO 01
27 RCL 1
28 R/S
29 RCL 0
30 f REG
31 STO 0
32 GTO 01
```

EXERCISE 13

Before starting store x_0 in R_0. When the program stops, the frequency for the event "All different" is displayed. Store 0 in R_5 and R_6 before the program is repeated.

```
01 1
02 STO +5
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 1
11 0
12 x
13 STO 4
14 f INT
15 STO 1
16 RCL 4
17 g FRAC
18 1
19 0
20 x
21 STO 4
22 f INT
23 STO 2
24 RCL 4
25 g FRAC
26 1
27 0
28 x
29 f INT
30 STO 3
31 RCL 2
32 f x=y
33 GTO 42
34 RCL 1
35 f x=y
36 GTO 42
37 RCL 3
38 f x=y
39 GTO 42
40 1
41 STO +6
42 RCL 5
43 f PAUSE
44 EEX
45 n
46 f x≠y
47 GTO 01
48 RCL 6
49 GTO 00
```

EXERCISE 14

Before starting store x_0 in R_0, 147 in R_4, and 10 in R_6. Store 0 in R_5 and R_7 before the program is repeated.

```
01 1
02 STO +5
03 RCL 0
04 RCL 4
05 x
06 g FRAC
07 STO 0
08 RCL 6
09 x
10 f INT
11 STO 1
12 RCL 0
13 RCL 4
14 x
15 g FRAC
16 STO 0
17 RCL 6
18 x
19 f INT
20 STO 2
21 RCL 0
22 RCL 4
23 x
24 g FRAC
25 STO 0
26 RCL 6
```

27 x
28 f INT
29 STO 3
30 RCL 2
31 $f\,x{=}y$
32 GTO 41
33 RCL 1
34 $f\,x{=}y$
35 GTO 41
36 RCL 3
37 $f\,x{=}y$
38 GTO 41
39 1
40 STO +7
41 RCL 5
42 f PAUSE
43 EEX
44 n
45 $f\,x{\neq}y$
46 GTO 01
47 RCL 7
48 GTO 00

EXERCISE 15

Before starting store x_0 in R_0 and 147 in R_7. Stop the program with the R/S-key when the pause indicates that a sufficient number of trials has been made. The frequency of the event is then recalled from R_4. Store 0 in R_3, R_4, and R_5 before the program is repeated.

01 RCL 0
02 RCL 7
03 x
04 g FRAC
05 STO 0
06 1
07 0
08 x
09 f INT
10 STO 1
11 1
12 STO +3
13 RCL 3
14 $f\,x{\neq}y$
15 GTO 19
16 RCL 1
17 STO 2
18 GTO 01
19 2
20 $f\,x{\neq}y$
21 GTO 27
22 RCL 1
23 RCL 2
24 $f\,x{=}y$
25 GTO 43
26 GTO 01
27 RCL 3
28 3
29 $f\,x{\neq}y$
30 GTO 37
31 RCL 2
32 RCL 1
33 $f\,x{=}y$
34 GTO 43
35 STO 2
36 GTO 01
37 RCL 1
38 RCL 2
39 $f\,x{=}y$
40 GTO 43
41 1
42 STO +4
43 1
44 STO +5
45 RCL 5
46 f PAUSE
47 0
48 STO 3
49 GTO 01

EXERCISE 16

Before starting store x_0 in R_0.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 1
09 0
10 x
11 f INT
12 g x=0
13 GTO 01
14 6
15 f x<y
16 GTO 01
17 ↓
18 f PAUSE
19 GTO 01
```

EXERCISE 17

Before starting store x_0 in R_0.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 6
09 x
10 f INT
11 1
12 +
13 f PAUSE
14 GTO 01
```

EXERCISE 19

Before starting the following programs, store x_0 in R_0.

(a)

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 6
11 x
12 f INT
13 1
14 +
15 f PAUSE
16 3
17 −
18 g x≠0
19 GTO 22
20 1
21 STO +2
22 RCL 2
23 RCL 1
24 ÷
25 f PAUSE
26 GTO 01
```

(b)

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 6
11 x
12 f INT
13 1
14 +
15 f PAUSE
16 2
17 f x<y
18 GTO 21
19 1
20 STO +2
21 RCL 2
22 RCL 1
23 ÷
24 f PAUSE
25 GTO 01
```

(c)

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 6
11 x
12 f INT
13 1
14 +
15 f PAUSE
16 2
17 ÷
18 g FRAC
19 g x=0
20 GTO 23
21 1
22 STO +2
23 RCL 2
24 RCL 1
25 ÷
26 f PAUSE
27 GTO 01
```

EXERCISE 20

Use the second program in Exercise 5 with the changes listed here. Change program step 06 to "*g* NOP" and program step 07 to "6." The tosses are then made with a die which gives 0, 1, 2, 3, 4, and 5 points.

EXERCISE 21

Before starting store x_0 in R_0. When the program stops, the mean is displayed. The next stop will display the standard deviation. The program may be repeated.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 6
09 x
10 f INT
11 1
12 +
13 Σ+
14 f PAUSE
15 ENTER
16 EEX
17 n
18 f x≠y
19 GTO 01
20 f x̄
21 R/S
22 f s
23 R/S
24 RCL 0
25 f REG
26 STO 0
27 GTO 01
```

EXERCISE 22

Before starting store x_0 in R_0. The first stop displays the mean; the second stop displays the standard deviation. By pressing the R/S-key you get the next values of the mean and the standard deviation.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 6
11 x
12 f INT
13 1
14 +
15 Σ+
16 ENTER
17 1
18 0
19 f x≠y
20 GTO 03
21 f x̄
22 R/S
23 f s
24 R/S
25 RCL 0
26 f REG
27 STO 0
28 RCL 1
29 EEX
30 2
31 f x≠y
32 GTO 01
33 GTO 00
```

EXERCISE 23

See the program in Exercise 21.

EXERCISE 24

See the program in Exercise 22.

EXERCISE 25

Before starting store x_0 in R_0.

01	RCL 0	19	x
02	1	20	f INT
03	4	21	2
04	7	22	+
05	x	23	+
06	g FRAC	24	f PAUSE
07	STO 0	25	7
08	6	26	$f\ x \geqslant y$
09	x	27	GTO 30
10	f INT	28	1
11	RCL 0	29	STO +2
12	1	30	1
13	4	31	STO +1
14	7	32	RCL 2
15	x	33	RCL 1
16	g FRAC	34	÷
17	STO 0	35	f PAUSE
18	6	36	GTO 01

EXERCISE 26

Before starting store x_0 in R_0 and 147 in R_7.

01	1	16	g FRAC
02	STO +1	17	STO 0
03	RCL 0	18	6
04	RCL 7	19	x
05	x	20	f INT
06	g FRAC	21	1
07	STO 0	22	+
08	6	23	$x \leftrightarrow y$
09	x	24	$f\ x \geqslant y$
10	f INT	25	GTO 28
11	1	26	1
12	+	27	STO +2
13	RCL 0	28	RCL 1
14	RCL 7	29	1
15	x	30	0

```
31  f x≠y
32  GTO 01
33  STO +3
34  RCL 2
35  f PAUSE
36  STO +4
37  RCL 4
38  RCL 3
39  ÷
40  f PAUSE
41  0
42  STO 1
43  STO 2
44  GTO 01
```

EXERCISE 27

Before starting store x_0 in R_0 and 147 in R_7.

```
01  1
02  STO +1
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  6
09  x
10  f INT
11  1
12  +
13  3
14  f x=y
15  GTO 29
16  RCL 0
17  RCL 7
18  x
19  g FRAC
20  STO 0
21  6
22  x
23  f INT
24  1
25  +
26  3
27  f x≠y
28  GTO 31
29  1
30  STO +2
31  RCL 1
32  1
33  0
34  f x≠y
35  GTO 01
36  STO +3
37  RCL 2
38  f PAUSE
39  STO +4
40  RCL 4
41  RCL 3
42  ÷
43  f PAUSE
44  0
45  STO 1
46  STO 2
47  GTO 01
```

EXERCISE 28

Before starting store x_0 in R_0. The following program makes 10 trials of the experiment to toss a die until a six is obtained. The number of tosses in each trial is displayed. The first stop gives the mean; the second gives the standard deviation. The number of trials can be changed to values up to 999. The program can be repeated.

```
01  0
02  STO 1
03  1
04  STO +1
05  RCL 0
06  1
07  4
08  7
09  x
10  g FRAC
11  STO 0
12  6
13  x
14  f INT
15  1
16  +
17  6
18  f x≠y
19  GTO 03
20  RCL 1
21  f PAUSE
22  Σ+
23  ENTER
24  1
25  0
26  g NOP
27  f x≠y
28  GTO 01
29  f x̄
30  R/S
31  f s
32  R/S
33  RCL 0
34  f REG
35  STO 0
36  GTO 03
```

EXERCISE 29

Before starting store x_0 in R_0 and 147 in R_7.

```
01  1
02  STO +1
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  6
09  x
10  f INT
11  1
12  +
13  RCL 0
14  RCL 7
15  x
16  g FRAC
17  STO 0
18  6
19  x
20  f INT
21  1
22  +
23  f x≥y
24  GTO 26
25  ↓
26  STO +2
27  f PAUSE
28  RCL 1
29  1
30  0
31  0
32  f x≠y
33  GTO 01
34  RCL 2
35  RCL 1
36  ÷
37  R/S
38  0
39  STO 1
40  STO 2
41  GTO 01
```

EXERCISE 30

Before starting store x_0 in R_0 and 147 in R_7. Before the program is repeated 0 must be stored in R_1, R_3, and R_4.

```
01 RCL 0
02 RCL 7
03 x
04 g FRAC
05 STO 0
06 6
07 x
08 f INT
09 1
10 STO +1
11 +
12 RCL 0
13 RCL 7
14 x
15 g FRAC
16 STO 0
17 6
18 x
19 f INT
20 1
21 STO +1
22 +
23 f x≥y
24 GTO 26
25 ↓
26 STO 2
27 RCL 1
28 m
29 f x=y
30 GTO 33
31 RCL 2
32 GTO 12
33 RCL 2
34 STO +3
35 1
36 STO +4
37 RCL 4
38 f PAUSE
39 EEX
40 n
41 f x=y
42 GTO 46
43 0
44 STO 1
45 GTO 01
46 RCL 3
47 x↔y
48 ÷
49 GTO 00
```

EXERCISE 31

Before starting store x_0 in R_0 and 147 in R_7.

```
01 RCL 0
02 RCL 7
03 x
04 g FRAC
05 STO 0
06 6
07 x
08 f INT
09 1
10 +
11 STO 1
12 RCL 0
13 RCL 7
14 x
15 g FRAC
16 STO 0
17 6
18 x
19 f INT
20 1
```

21 +
22 STO 2
23 RCL 1
24 3
25 $f\ x \geqslant y$
26 GTO 32
27 RCL 1
28 RCL 2
29 $f\ x < y$
30 GTO 37
31 GTO 39
32 RCL 2
33 RCL 1
34 $f\ x < y$
35 GTO 37
36 GTO 39
37 1
38 STO +3
39 1
40 STO +4
41 RCL 4
42 *f* PAUSE
43 EEX
44 *n*
45 $f\ x \neq y$
46 GTO 01
47 RCL 3
48 RCL 4
49 ÷

EXERCISE 32

(a)
01 RCL 0
02 1
03 4
04 7
05 *x*
06 *g* FRAC
07 STO 0
08 2
09 *x*
10 *f* INT
11 *f* PAUSE
12 GTO 01
13 GTO 01

(b)
01 RCL 0
02 1
03 3
04 7
05 *x*
06 *g* FRAC
07 STO 0
08 ·
09 5
10 +
11 *f* INT
12 *f* PAUSE

In both programs x_0 is stored in R_0 before starting.

EXERCISE 33

Before starting store x_0 in R_0 and the probability p in R_1.

01 1
02 STO + 2
03 RCL 0
04 1
05 4
06 7
07 *x*
08 *g* FRAC
09 STO 0
10 RCL 1
11 +
12 *f* INT
13 *f* PAUSE
14 STO + 3

15 RCL 2
16 *f* PAUSE
17 EEX
18 *n*
19 $f\,x \neq y$
20 GTO 01
21 RCL 3
22 R/S
23 0
24 STO 2
25 STO 3
26 GTO 01

EXERCISE 34

Before starting store x_0 in R_0 and the probability p in R_1. After execution of the program the R/S-key will clear all registers except R_0. Store p value in R_1 before the program is repeated.

01 0
02 STO 2
03 1
04 STO +2
05 RCL 0
06 1
07 4
08 7
09 x
10 *g* FRAC
11 STO 0
12 RCL 1
13 +
14 *f* INT
15 $g\,x=0$
16 GTO 03
17 RCL 2
18 *f* PAUSE
19 $\Sigma +$
20 ENTER
21 *f* PAUSE
22 EEX
23 *n*
24 $f\,x \neq y$
25 GTO 01
26 $f\,\overline{x}$
27 R/S
28 $f\,s$
29 R/S
30 RCL 0
31 *f* REG
32 STO 0
33 GTO 00

EXERCISE 35

Before starting store x_0 in R_0. When the program stops, the number of ones is displayed. Pressing the R/S-key will start a new sequence of 10 trials.

01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 *g* FRAC
09 STO 0
10 ·
11 4
12 +
13 *f* INT
14 *f* PAUSE
15 STO +2
16 RCL 1

```
17 1
18 0
19 f x y
20 GTO 01
21 RCL 2
22 R/S
23 RCL 0
24 f REG
25 ST 0
26 GTO 01
```

EXERCISE 36

Before starting store x_0 in R_0 and p in R_7. The first stop displays the frequency for 0 0; the second stop displays the frequency for the event 0 1 or 1 0. After execution of the program the R/S-key will clear all registers except R_0. Store p in R_7 before the program is repeated.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 RCL 7
11 +
12 f INT
13 STO +2
14 RCL 1
15 2
16 f x≠y
17 GTO 01
18 1
19 STO +5
20 RCL 2
21 g x=0
22 GTO 30
23 1
24 −
25 g x≠0
26 GTO 32
27 1
28 STO +4
29 GTO 32
30 1
31 STO +3
32 0
33 STO 1
34 STO 2
35 RCL 5
36 f PAUSE
37 EEX
38 n
39 f x≠y
40 GTO 01
41 RCL 3
42 R/S
43 RCL 4
44 R/S
45 RCL 0
46 f REG
47 STO 0
48 GTO 00
```

EXERCISE 37

Before starting store x_0 in R_0, p_A in R_6, and p_B in R_7. The program can be repeated.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 RCL 6
09 +
10 f INT
11 STO 2
12 RCL 0
13 1
14 4
15 7
16 x
17 g FRAC
18 STO 0
19 RCL 7
20 +
21 f INT
22 g x≠0
23 GTO 28
24 RCL 2
25 g x=0
26 GTO 01
27 STO +3
28 STO +4
29 RCL 4
30 f PAUSE
31 EEX
32 n
33 f x≠y
34 GTO 01
35 RCL 3
36 RCL 4
37 ÷
38 R/S
39 0
40 STO 3
41 STO 4
42 GTO 01
```

EXERCISE 38

(a) Before starting store x_0 in R_0.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 2
09 x
10 f INT
11 2
12 x
13 1
14 —
15 f PAUSE
16 GTO 01
```

(b) Before starting store x_0 in R_0 and p in R_7.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 RCL 7
09 +
10 f INT
11 2
12 x
13 1
14 —
15 f PAUSE
16 GTO 01
```

EXERCISE 39

(a) Before starting store x_0 in R_0. The three stops display, in order, the frequencies of ones, twos, and threes. The program can be repeated.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 3
09 x
10 f INT
11 g x=0
12 GTO 22
13 1
14 −
15 g x=0
16 GTO 19
17 STO +3
18 GTO 24
19 1
20 STO +2
21 GTO 24
22 1
23 STO +1
24 1
25 STO +4
26 RCL 4
27 f PAUSE
28 EEX
29 n
30 f x≠y
31 GTO 01
32 RCL 1
33 R/S
34 RCL 2
35 R/S
36 RCL 3
37 R/S
38 RCL 0
39 f REG
40 STO 0
41 GTO 01
```

(b) Before starting store x_0 in R_0, 0.5 in R_6, and 0.25 in R_7. The program can be repeated.

```
01 RCL 0
02 1
03 4
04 7
05 x
06 g FRAC
07 STO 0
08 RCL 6
09 +
10 f INT
11 RCL 0
12 RCL 7
13 +
14 f INT
15 +
16 g x=0
17 GTO 27
18 1
19 −
20 g x=0
21 GTO 24
22 STO +3
23 GTO 29
24 1
25 STO +2
26 GTO 29
27 1
28 STO +1
```

```
29  1
30  STO +4
31  RCL 4
32  f PAUSE
33  EEX
34  n
35  f x≠y
36  GTO 01
37  RCL 1
38  R/S
39  RCL 2
40  R/S
41  RCL 3
42  R/S
43  0
44  STO 1
45  STO 2
46  STO 3
47  STO 4
48  GTO 01
```

EXERCISES 40–42

The following program will simulate the Markov chain in Figure 11. The program can be used in Exercise 40 with $a = 0.4$ and $b = 0.6$, and in Exercise 41 with $a = 0.6$ and $b = 0.4$. Before starting store x_0 in R_0, a in R_1, b in R_2, and the initial state (0 or 1) in R_3.

```
01  RCL 0
02  1
03  4
04  7
05  x
06  g FRAC
07  STO 0
08  RCL 3
09  g x=0
10  GTO 18
11  RCL 0
12  RCl 2
13  +
14  f INT
15  R/S
16  STO 3
17  GTO 01
18  RCL 0
19  RCL 1
20  +
21  f INT
22  R/S
23  STO 3
24  GTO 01
```

EXERCISE 43

Before starting store x_0 in R_0, a in R_1, b in R_2, the initial state y_0 in R_3, and $1 - y_0$ in R_4.

```
01  RCL 0
02  1
03  4
04  7
05  x
06  g FRAC
07  STO 0
08  RCL 1
09  +
10  RCL 4
11  x
12  RCL 0
13  RCL 2
14  +
```

```
15 RCL 3
16 x
17 +
18 f INT
19 f PAUSE
20 STO 3
21 CHS
22 1
23 +
24 STO 4
25 GTO 01
```

The formula used in the program above is equivalent to the formula given in the exercise.

EXERCISE 44

Before starting store x_0 in R_0, 10 in R_6, and 147 in R_7.

```
01 0
02 STO 3
03 STO 4
04 1
05 STO +4
06 RCL 0
07 RCL 7
08 x
09 g FRAC
10 STO 0
11 RCL 6
12 x
13 f INT
14 STO 1
15 RCL 0
16 RCL 7
17 x
18 g FRAC
19 STO 0
20 RCL 6
21 x
22 f INT
23 STO 2
24 RCL 0
25 RCL 7
26 x
27 g FRAC
28 STO 0
29 RCL 6
30 x
31 f INT
32 RCL 2
33 x↔y
34 f x≥y
35 GTO 42
36 x↔y
37 RCL 1
38 f x≥y
39 GTO 42
40 1
41 STO +3
42 RCL 4
43 f PAUSE
44 EEX
45 n
46 f x≠y
47 GTO 04
48 RCL 3
49 GTO 00
```

EXERCISE 45

Before starting store x_0 in R_0. The program can be repeated.

```
01 1
02 STO +1
03 RCL 1
04 7
05 f x=y
06 GTO 24
07 RCL 0
08 1
09 4
10 7
11 x
12 g FRAC
13 STO 0
14 6
15 x
16 f INT
17 1
18 +
19 RCL 1
20 f x≠y
21 GTO 01
22 1
23 STO +2
24 1
25 STO +3
26 RCL 3
27 f PAUSE
28 EEX
29 n
30 f x=y
31 GTO 35
32 0
33 STO 1
34 GTO 01
35 RCL 2
36 R/S
37 RCL 0
38 f REG
39 STO 0
40 GTO 01
```

EXERCISE 46

Before starting store x_0 in R_0 and 147 in R_7. When the program stops, the frequency for no coincidences can be recalled from R_3, the frequency for one coincidence from R_4, and the frequency for more than one coincidence from R_5. Before the program is repeated, 0 must be stored in R_1–R_6.

```
01 1
02 STO +1
03 RCL 1
04 7
05 f x=y
06 GTO 23
07 RCL 0
08 RCL 7
09 x
10 g FRAC
11 STO 0
12 6
13 x
14 f INT
15 1
16 +
17 RCL 1
18 f x≠y
19 GTO 01
20 1
21 STO +2
22 GTO 01
23 RCL 2
24 g x≠0
25 GTO 29
26 1
27 STO +3
28 GTO 38
```

```
29  1
30  –
31  g x≠0
32  GTO 36
33  1
34  STO +4
35  GTO 38
36  1
37  STO +5
38  0
39  STO 1
40  STO 2
41  1
42  STO +6
43  RCL 6
44  f PAUSE
45  EEX
46  n
47  f x≠y
48  GTO 01
49  GTO 00
```

EXERCISE 47

Before starting store x_0 in R_0 and n in R_7.

```
01  1
02  STO +1
03  RCL 7
04  1
05  +
06  RCL 1
07  f x=y
08  GTO 26
09  RCL 0
10  1
11  4
12  7
13  x
14  g FRAC
15  STO 0
16  RCL 7
17  x
18  f INT
19  1
20  +
21  RCL 1
22  f x≠y
23  GTO 01
24  1
25  STO +2
26  1
27  STO +3
28  RCL 3
29  f PAUSE
30  EEX
31  m
32  f x=y
33  GTO 37
34  0
35  STO 1
36  GTO 01
37  RCL 2
38  R/S
39  0
40  STO 1
41  STO 2
42  STO 3
43  GTO 01
```

EXERCISE 48

Before starting store x_0 in R_0, n in R_6, and 147 in R_7. The first stop displays the frequency for no coincidences, and the second the frequency for one coincidence. Store 0 in R_1–R_5 before repeating.

```
01  1
02  STO +1
03  RCL 6
04  +
05  RCL 1
06  f x=y
07  GTO 24
08  RCL 0
09  RCL 7
10  x
11  g FRAC
12  STO 0
13  RCL 6
14  x
15  f INT
16  1
17  +
18  RCL 1
19  f x≠y
20  GTO 01
21  1
22  STO +2
23  GTO 01
24  RCL 2
25  g x≠0
26  GTO 30
27  1
28  STO +3
29  GTO 36
30  1
31  –
32  g x≠0
33  GTO 36
34  1
35  STO +4
36  1
37  STO +5
38  0
39  STO 1
40  STO 2
41  RCL 5
42  f PAUSE
43  EEX
44  m
45  f x≠y
46  GTO 01
47  RCL 3
48  R/S
49  RCL 4
```

EXERCISE 49

Before starting store x_0 in R_0. When the program stops, the number of white marbles in the sample is displayed. The program can be repeated.

```
01  m
02  g NOP
03  STO 1
04  n
05  g NOP
06  STO 2
07  RCL 0
08  1
09  4
10  7
11  x
12  g FRAC
13  STO 0
14  RCL 1
15  RCL 2
16  +
17  RCL 2
18  x⇌y
19  ÷
20  RCL 0
21  +
22  f INT
23  f PAUSE
24  STO +3
```

```
25 g x=0
26 GTO 29
27 STO −2
28 GTO 31
29 1
30 STO −1
31 1
32 STO +4
33 RCL 4
34 r
35 g NOP
36 f x≠y
37 GTO 07
38 RCL 3
39 R/S
40 0
41 STO 3
42 STO 4
43 GTO 01
```

EXERCISE 50

Before starting store x_0 in R_0. Store 0 in R_3 and R_5 before the program is repeated.

```
01 m
02 g NOP
03 STO 1
04 n
05 g NOP
06 STO 2
07 RCL 0
08 1
09 4
10 7
11 x
12 g FRAC
13 STO 0
14 RCL 1
15 RCL 2
16 +
17 RCL 2
18 x⇌y
19 ÷
20 RCL 0
21 +
22 f INT
23 STO +3
24 g x=0
25 GTO 28
26 STO −2
27 GTO 30
28 1
29 STO −1
30 1
31 STO +4
32 RCL 4
33 r
34 g NOP
35 f x≠y
36 GTO 07
37 0
38 STO 4
39 1
40 STO +5
41 RCL 5
42 f PAUSE
43 EEX
44 s
45 f x≠y
46 GTO 01
47 RCL 3
48 GTO 00
```

EXERCISE 51

Before starting store x_0 in R_0, m in R_1, and n in R_2. The program can be repeated.

```
01 RCL 1
02 STO 7
03 1
04 STO +3
05 RCL 0
06 1
07 4
08 7
09 x
10 g FRAC
11 STO 0
12 RCL 1
13 RCL 2
14 +
15 RCL 2
16 x↔y
17 ÷
18 RCL 0
19 +
20 f INT
21 f PAUSE
22 g x≠0
23 GTO 27
24 1
25 STO −1
26 GTO 03
27 RCL 3
28 R/S
29 0
30 STO 3
31 RCL 7
32 STO 1
33 GTO 01
```

EXERCISE 52

Before starting store x_0 in R_0. When the program stops, the total number of marbles taken in the trials is displayed. The mean is found by division by 10^s. Before the program is repeated store 0 in R_3, R_4, and R_5.

```
01 m
02 g NOP
03 g NOP
04 STO 1
05 n
06 g NOP
07 g NOP
08 STO 2
09 1
10 STO +3
11 RCL 0
12 1
13 4
14 7
15 x
16 g FRAC
17 STO 0
18 RCL 1
19 RCL 2
20 +
21 RCL 2
22 x y
23
24 RCL 0
25 +
26 f INT
27 g x 0
28 GTO 32
29 1
30 STO −1
31 GTO 09
32 RCL 3
33 STO +4
34 1
35 STO +5
36 RCL 5
37 f PAUSE
38 EEX
```

```
39  s
40  f x=y
41  GTO 45
42  0
43  STO 3
44  GTO 01
45  RCL 4
46  GTO 00
47
```

EXERCISE 53

Before starting store x_0 in R_0, m in R_1, and n in R_2. During the pause, the remaining number of white marbles is displayed. When the program stops, the number of marbles taken out of the urn is displayed. The program can be repeated.

```
01  RCL 1
02  STO 6
03  RCL 2
04  STO 7
05  RCL 0
06  1
07  4
08  7
09  x
10  g FRAC
11  STO 0
12  RCL 1
13  RCL 2
14  +
15  RCL 2
16  x↔y
17  ÷
18  RCL 0
19  +
20  f INT
21  g x=0
22  GTO 25
23  STO −2
24  GTO 27
25  1
26  STO −1
27  1
28  STO +3
29  RCL 2
30  f PAUSE
31  g x≠0
32  GTO 05
33  RCL 3
34  R/S
35  RCL 6
36  STO 1
37  RCL 7
38  STO 2
39  0
40  STO 3
41  GTO 05
```

EXERCISE 54

Before starting store x_0 in R_0, m in R_1, and n in R_2.

```
01  RCL 0
02  1
03  4
04  7
05  x
06  g FRAC
07  STO 0
08  RCL 1
09  RCL 2
10  +
11  RCL 2
12  x↔y
```

```
13 ÷
14 RCL 0
15 +
16 f INT
17 g x=0
18 GTO 21
19 STO −2
20 GTO 23
21 1
22 STO −1
23 1
24 STO +3
25 RCL 2
26 g x≠0
27 GTO 01
28 RCL 3
29 STO +4
30 1
31 STO +5
32 0
33 STO 3
34 m
35 STO 1
36 n
37 STO 2
38 RCL 5
39 f PAUSE
40 EEX
41 n
42 f x≠y
43 GTO 01
44 RCL 4
45 R/S
```

EXERCISE 55

Before starting store x_0 in R_0, m in R_6, and n in R_7. Concerning program steps 22 and 24, see Exercise 58. The program can be repeated.

```
01 RCL 6
02 STO 1
03 RCL 7
04 STO 2
05 1
06 STO +3
07 RCL 0
08 1
09 4
10 7
11 x
12 g FRAC
13 STO 0
14 2
15 x
16 f INT
17 g x=0
18 GTO 22
19 STO −2
20 g NOP
21 GTO 25
22 1
23 STO −1
24 g NOP
25 RCL 1
26 f PAUSE
27 RCL 2
28 f PAUSE
29 x
30 g x≠0
31 GTO 05
32 RCL 3
33 R/S
34 0
35 STO 3
36 GTO 01
```

EXERCISE 56

Before starting store x_0 in R_0, m in R_6, and n in R_7. Concerning program steps 20 and 24, see Exercise 59. The program can be repeated.

```
01  RCL 6
02  STO 1
03  RCL 7
04  STO 2
05  1
06  STO +3
07  RCL 0
08  1
09  4
10  7
11  x
12  g FRAC
13  STO 0
14  2
15  x
16  f INT
17  g x=0
18  GTO 22
19  STO −2
20  g NOP
21  GTO 25
22  1
23  STO −1
24  g NOP
25  RCL 1
26  RCL 2
27  x
28  g x≠0
29  GTO 05
30  1
31  STO +4
32  RCL 4
33  f PAUSE
34  EEX
35  s
36  f x≠y
37  GTO 01
38  RCL 3
39  EEX
40  s
41  ÷
42  R/S
43  0
44  STO 3
45  STO 4
46  GTO 01
```

EXERCISE 57

Before starting store x_0 in R_0, m in R_6, and n in R_7. When the program stops, the total number of times the smaller tower was destroyed is displayed. The relative frequency is obtained by division by 10^s. The program can be repeated.

```
01  RCL 6
02  STO 1
03  RCL 7
04  STO 2
05  RCL 0
06  1
07  4
08  7
09  x
10  g FRAC
11  STO 0
12  2
13  x
14  f INT
15  g x=0
16  GTO 20
```

```
17  STO −2
18  g NOP
19  GTO 23
20  1
21  STO −1
22  g NOP
23  RCL 1
24  RCL 2
25  x
26  g x≠0
27  GTO 05
28  RCL 2
29  g x≠0
30  GTO 33
31  1
32  STO +3
33  1
34  STO +4
35  RCL 4
36  f PAUSE
37  EEX
38  s
39  f x≠y
40  GTO 01
41  RCL 3
42  R/S
43  0
44  STO 3
45  STO 4
46  GTO 01
47
```

Concerning program steps 18 and 22, see Exercise 60.

EXERCISE 58

Use the program in Exercise 55 but change program step 20 to "STO +1" and program step 24 to "STO + 2."

EXERCISE 59

Use the program in Exercise 56 but change program step 20 to "STO +1" and program step 24 to "STO +2."

EXERCISE 60

Use the program in Exercise 57 but change program step 18 to "STO +1" and program step 22 to "STO +2."

EXERCISE 61

Before starting store x_0 in R_0 and 147 in R_7. The program displays the successive number of blocks in the towers. The program can be repeated.

```
01  m
02  g NOP
03  STO 1
04  n
05  g NOP
06  STO 2
07  r
08  g NOP
09  STO 3
10  1
11  STO +4
12  RCL 0
```

```
13 RCL 7
14 x
15 g FRAC
16 STO 0
17 3
18 x
19 f INT
20 g x=0
21 GTO 28
22 1
23 −
24 g x=0
25 GTO 31
26 STO −3
27 GTO 33
28 1
29 STO −1
30 GTO 33
31 1
32 STO −2
33 RCL 1
34 f PAUSE
35 RCL 2
36 f PAUSE
37 x
38 RCL 3
39 f PAUSE
40 x
41 g x≠0
42 GTO 10
43 RCL 4
44 R/S
45 0
46 STO 4
47 GTO 01
```

EXERCISE 62

Before starting store x_0 in R_0 and 147 in R_7. When the program stops, the total number of blocks taken from the towers is displayed. The mean is obtained by division by 10^3. The program can be repeated.

```
01 m
02 STO 1
03 n
04 STO 2
05 r
06 STO 3
07 1
08 STO +4
09 RCL 0
10 RCL 7
11 x
12 g FRAC
13 STO 0
14 3
15 x
16 f INT
17 g x=0
18 GTO 25
19 1
20 −
21 g x=0
22 GTO 28
23 STO −3
24 GTO 30
25 1
26 STO −1
27 GTO 30
28 1
29 STO −2
30 RCL 1
31 RCL 2
32 x
33 RCL 3
34 x
35 g x≠0
36 GTO 07
```

```
37  1
38  STO +5
39  RCL 5
40  f PAUSE
41  EEX
42  s
43  f x≠y
44  GTO 01
45  RCL 4
46  R/S
47  0
48  STO 4
49  STO 5
```

EXERCISE 63

Before starting store x_0 in R_0. The first stop displays the total number of blocks taken from the towers; the second stop displays the number of blocks remaining in the first tower. The program can be repeated.

```
01  8
02  STO 1
03  6
04  STO 2
05  4
06  STO 3
07  1
08  STO +4
09  RCL 0
10  RCL 7
11  x
12  g FRAC
13  STO 0
14  3
15  x
16  f INT
17  g x=0
18  GTO 25
19  1
20  −
21  g x=0
22  GTO 28
23  STO −3
24  GTO 30
25  1
26  STO −1
27  GTO 30
28  1
29  STO −2
30  RCL 1
31  f PAUSE
32  RCL 2
33  f PAUSE
34  x
35  RCL 3
36  f PAUSE
37  x
38  g x≠0
39  GTO 07
40  RCL 4
41  R/S
42  RCL 1
43  R/S
44  0
45  STO 4
46  GTO 01
```

EXERCISE 64

Before starting store x_0 in R_0, m in R_1, n in R_2, r in R_3, -1 in R_6, and 147 in R_7. When the program stops, the total number of moved blocks can be recalled from R_4. Before the program is started again, store m in R_1, n in R_2, r in R_3, and 0 in R_4.

In the program a spinner with the outcomes 0, 1, 2, 3, 4, and 5 is spun. To identify the outcome, 1 is subtracted, and we determine the difference is negative. Access to the chain of program steps mentioned in the commentary will be with −1. That's the reason why all signs have been changed in this chain.

```
01  RCL 0          25  GTO 35
02  RCL 7          26  1
03  x              27  −
04  g FRAC         28  g x<0
05  STO 0          29  GTO 33
06  6              30  RCL 6
07  x              31  STO −1
08  f INT          32  STO +2
09  1              33  STO −3
10  STO +4         34  STO +2
11  −              35  STO −3
12  g x<0          36  STO +1
13  GTO 41         37  STO +1
14  1              38  STO −2
15  −              39  STO −3
16  g x<0          40  STO −2
17  GTO 39         41  STO −1
18  1              42  STO +2
19  −              43  RCL 1
20  g x<0          44  RCL 2
21  GTO 37         45  x
22  1              46  RCL 3
23  −              47  x
24  g x<0          48  g x≠0
                   49  GTO 01
```

EXERCISE 65

Before starting store x_0 in R_0, −1 in R_5, and 147 in R_7. In this program one block is first placed on one tower, and then a block is removed from a tower (see the key program steps 10, 11). Each time the number of blocks in the tower is displayed. The program can be repeated.

```
01  m
02  STO 1
03  n
04  STO 2
05  r
06  STO 3
07  1
08  STO +4
09  1
10  CHS
11  STO x5
12  RCL 0
13  RCL 7
14  x
15  g FRAC
16  STO 0
17  3
18  x
19  f INT
20  g x=0
21  GTO 32
22  1
23  —
24  g x=0
25  GTO 29
26  RCL 5
27  STO +3
28  GTO 34
29  RCL 5
30  STO +2
31  GTO 34
32  RCL 5
33  STO +1
34  RCL 1
35  f PAUSE
36  RCL 2
37  f PAUSE
38  x
39  RCL 3
40  f PAUSE
41  x
42  g x≠0
43  GTO 07
44  RCL 4
45  2
46  ÷
47  R/S
48  0
49  STO 4
```

EXERCISE 66

Before starting store x_0 in R_0, -1 in R_5, and 147 in R_7. When the program stops, divide by 2.10^s to obtain the mean number of moves.

```
01  m
02  STO 1
03  n
04  STO 2
05  r
06  STO 3
07  1
08  STO +4
09  1
10  CHS
11  STO x5
12  RCL 0
13  RCL 7
14  x
15  g FRAC
16  STO 0
17  3
18  x
19  f INT
20  g x=0
21  GTO 32
22  1
23  —
24  g x=0
```

25 GTO 29
26 RCL 5
27 STO +3
28 GTO 34
29 RCL 5
30 STO +2
31 GTO 34
32 RCL 5
33 STO +1
34 RCL 1
35 RCL 2
36 x
37 RCL 3
38 x
39 $g\,x\ 0$
40 GTO 07
41 1
42 STO +6
43 RCL 6
44 f PAUSE
45 EEX
46 s
47 $f\,x\ y$
48 GTO 01
49 RCL 4

EXERCISE 68

Before starting store x_0 in R_0. The players are denoted 1 and -1, as are their colors also. The first stop indicates the winning player and the second the duration of the game.

01 1
02 CHS
03 STO 6
04 RCL 6
05 CHS
06 STO 6
07 RCL 0
08 1
09 4
10 7
11 x
12 g FRAC
13 STO 0
14 2
15 x
16 f INT
17 $g\,x{=}0$
18 GTO 22
19 RCL 6
20 STO 2
21 GTO 24
22 RCL 6
23 STO 1
24 1
25 STO +3
26 RCL 1
27 f PAUSE
28 RCL 2
29 f PAUSE
30 $f\,x{\neq}y$
31 GTO 04
32 RCL 1
33 R/S
34 RCL 3
35 R/S
36 RCL 0
37 f REG
38 STO 0
39 GTO 01

EXERCISE 69

Before starting store x_0 in R_0. The first stop will display the total number of moves. The mean is found by division by 10^n. The second stop will display the number of times that the first player was the winner. Before repetition, recall R_0, clear the registers, and store the number in the x-register back into R_0.

```
01 1
02 CHS
03 STO 6
04 RCL 6
05 CHS
06 STO 6
07 1
08 STO +3
09 RCL 0
10 1
11 4
12 7
13 x
14 g FRAC
15 STO 0
16 2
17 x
18 f INT
19 g x=0
20 GTO 24
21 RCL 6
22 STO 2
23 GTO 26
24 RCL 6
25 STO 1
26 RCL 1
27 RCL 2
28 f x≠y
29 GTO 04
30 RCL 1
31 g x<0
32 GTO 35
33 1
34 STO +4
35 1
36 STO +5
37 RCL 5
38 f PAUSE
39 EEX
40 n
41 f x=y
42 GTO 47
43 0
44 STO 1
45 STO 2
46 GTO 01
47 RCL 3
48 R/S
49 RCL 4
```

EXERCISE 70

Before starting store x_0 in R_0 and 147 in R_7. The program displays the colors of the uppermost blocks in the first two towers and, if these are equal, also the color of the uppermost block on the third tower. The first will display the winner and the second the duration of the game.

```
01 0
02 STO 1
03 STO 2
04 STO 3
05 STO 4
06 1
07 CHS
08 STO 6
09 RCL 6
10 CHS
11 STO 6
12 RCL 0
```

```
13 RCL 7
14 x
15 g FRAC
16 STO 0
17 3
18 x
19 f INT
20 g x=0
21 GTO 32
22 1
23 −
24 g x=0
25 GTO 29
26 RCL 6
27 STO 3
28 GTO 34
29 RCL 6
30 STO 2
31 GTO 34
32 RCL 6
33 STO 1
34 1
35 STO +4
36 RCL 1
37 f PAUSE
38 RCL 2
39 f PAUSE
40 f x≠y
41 GTO 09
42 RCL 3
43 f PAUSE
44 f x≠y
45 GTO 09
46 RCL 1
47 R/S
48 RCL 4
49 GTO 00
```

EXERCISE 71

Before starting store x_0 in R_0 and 147 in R_7. When the program stops, recall register R_4 and divide by 10^n to obtain an estimate of the probability that the second player wins. Before repetition, 0 must be stored in R_4 and R_5.

```
01 1
02 CHS
03 STO 6
04 RCL 6
05 CHS
06 STO 6
07 RCL 0
08 RCL 7
09 x
10 g FRAC
11 STO 0
12 3
13 x
14 f INT
15 g x=0
16 GTO 27
17 1
18 −
19 g x=0
20 GTO 24
21 RCL 6
22 STO 3
23 GTO 29
24 RCL 6
25 STO 2
26 GTO 29
27 RCL 6
28 STO 1
29 RCL 1
30 RCL 2
31 f x≠y
32 GTO 04
33 RCL 3
34 f x≠y
```

35 GTO 04
36 *g x*≥0
37 GTO 40
38 1
39 STO +4
40 STO +5
41 RCL 5
42 *f* PAUSE
43 EEX
44 *n*
45 STO 1
46 STO 2
47 *f x*=*y*
48 GTO 00
49 GTO 01

EXERCISE 72

Before starting store x_0 in R_0 and 147 in R_7. When the program stops, recall register R_4 and divide by 10^n to get the average duration. Clear R_3, R_4, and R_5 before repetition.

01 0
02 STO 1
03 STO 2
04 1
05 CHS
06 STO 6
07 RCL 6
08 CHS
09 STO 6
10 RCL 0
11 RCL 7
12 *x*
13 *g* FRAC
14 STO 0
15 3
16 *x*
17 *f* INT
18 *g x*=0
19 GTO 30
20 1
21 –
22 *g x*=0
23 GTO 27
24 RCL 6
25 STO 3
26 GTO 32
27 RCL 6
28 STO 2
29 GTO 32
30 RCL 6
31 STO 1
32 1
33 STO +4
34 RCL 1
35 RCL 2
36 *f x*≠*y*
37 GTO 07
38 RCL 3
39 *f x*≠*y*
40 GTO 07
41 1
42 STO +5
43 RCL 5
44 *f* PAUSE
45 EEX
46 *n*
47 *f x*=*y*
48 GTO 00
49 GTO 01

EXERCISE 73

The following program will play the game with two players and four towers. Before starting store x_0 in R_0, -1 in R_6, and 147 in R_7. When the program stops, the duration is displayed. Clear $R_1 - R_5$ before repetition, and also store -1 in R_6.

```
01 RCL 6
02 CHS
03 STO 6
04 RCL 0
05 RCL 7
06 x
07 g FRAC
08 STO 0
09 4
10 x
11 f INT
12 g x=0
13 GTO 31
14 1
15 –
16 g x=0
17 GTO 28
18 1
19 –
20 g x=0
21 GTO 25
22 RCL 6
23 STO 4
24 GTO 33
25 RCL 6
26 STO 3
27 GTO 33
28 RCL 6
29 STO 2
30 GTO 33
31 RCL 6
32 STO 1
33 1
34 STO +5
35 RCL 1
36 f PAUSE
37 RCL 2
38 f PAUSE
39 f x≠y
40 GTO 01
41 RCL 3
42 f PAUSE
43 f x≠y
44 GTO 01
45 RCL 4
46 f PAUSE
47 f x≠y
48 GTO 01
49 RCL 5
```

EXERCISE 74

Before starting store x_0 in R_0 and 147 in R_7. The first stop will display the winning player and the second stop the duration. The program can be repeated.

```
01 1
02 STO +3
03 RCL 3
04 4
05 –
06 g x≠0
07 GTO 11
08 0
09 STO 3
10 GTO 01
11 RCL 0
12 RCL 7
13 x
14 g FRAC
15 STO 0
16 2
17 x
18 f INT
```

```
19 g x=0
20 GTO 24
21 RCL 3
22 STO 2
23 GTO 26
24 RCL 3
25 STO 1
26 1
27 STO +4
28 RCL 1
29 f PAUSE
30 RCL 2
31 f PAUSE
32 f x≠y
33 GTO 01
34 RCL 1
35 R/S
36 RCL 4
37 R/S
38 0
39 STO 1
40 STO 2
41 STO 3
42 STO 4
43 GTO 01
```

EXERCISE 75

Before starting store x_0 in R_0, 10^n (or any desired number of games) in R_6, and 147 in R_7. When the program stops, the number of times the first player won is displayed. Clear $R_1 - R_5$ before repetition. Change program step 33 to "2" or "3" to get the frequency for Players 2 and 3.

```
01 1
02 STO +3
03 RCL 3
04 4
05 –
06 g x≠0
07 GTO 11
08 0
09 STO 3
10 GTO 01
11 RCL 0
12 RCL 7
13 x
14 g FRAC
15 STO 0
16 2
17 x
18 f INT
19 g x=0
20 GTO 24
21 RCL 3
22 STO 2
23 GTO 26
24 RCL 3
25 STO 1
26 RCL 1
27 RCL 2
28 f x≠y
29 GTO 01
30 1
31 STO +4
32 RCL 1
33 1
34 –
35 g x≠0
36 GTO 39
37 1
38 STO +5
39 RCL 4
40 f PAUSE
41 RCL 6
42 f x=y
```

43 GTO 49
44 0
45 STO 1
46 STO 2
47 STO 3
48 GTO 01
49 RCL 5

EXERCISE 76

Before starting store x_0 in R_0 and 147 in R_7. When the program stops, the total number of blocks placed is displayed. Divide by 10^n to obtain the average duration. Clear R_1–R_5 before repetition.

01 1
02 STO +3
03 RCL 3
04 4
05 −
06 $g\ x \neq 0$
07 GTO 11
08 0
09 STO 3
10 GTO 01
11 RCL 0
12 RCL 7
13 x
14 g FRAC
15 STO 0
16 2
17 x
18 f INT
19 $g\ x=0$
20 GTO 24
21 RCL 3
22 STO 2
23 GTO 26
24 RCL 3
25 STO 1
26 1
27 STO +4
28 RCL 1
29 RCL 2
30 $f\ x \neq y$
31 GTO 01
32 1
33 STO +5
34 RCL 5
35 f PAUSE
36 EEX
37 n
38 $f\ x=y$
39 GTO 45
40 0
41 STO 1
42 STO 2
43 STO 3
44 GTO 01
45 RCL 4
46 GTO 00

EXERCISE 77

Change program step 04 in the programs for Exercises 74–76 to "$n-1$" instead of "4" to obtain a game with $n-1$ players.

EXERCISE 78

Before starting store x_0 in R_0 and 147 in R_7. The first stop will display the duration of the game. If, after the second stop, 147 is stored in R_7, the program can be repeated.

```
01 1
02 STO +4
03 RCL 4
04 4
05 −
06 g x≠0
07 GTO 11
08 0
09 STO 4
10 GTO 01
11 RCL 0
12 RCL 7
13 x
14 g FRAC
15 STO 0
16 3
17 x
18 f INT
19 g x=0
20 GTO 31
21 1
22 −
23 g x=0
24 GTO 28
25 RCL 4
26 STO 3
27 GTO 33
28 RCL 4
29 STO 2
30 GTO 33
31 RCL 4
32 STO 1
33 1
34 STO +5
35 RCL 1
36 f PAUSE
37 RCL 2
38 f PAUSE
39 f x≠y
40 GTO 01
41 RCL 3
42 f PAUSE
43 f x≠y
44 GTO 01
45 RCL 5
46 R/S
47 RCL 0
48 f REG
49 STO 0
```

If program step 04 is changed to "n" the program will play the game with $n-1$ players and 3 towers.

EXERCISE 79

Before starting store x_0 in R_0, 2 in R_1 and R_2, -1 in R_6, and 147 in R_7. If 2 is not stored in R_1 and R_2, two diagonal squares would have the same number 0 after the first move. When the program stops, the total number of moves is displayed. Before the program can be repeated, the registers R_3, R_4, and R_5 must be cleared. Furthermore, 2 must be stored in R_1 and R_2, and -1 must be stored in R_6.

```
01 RCL 6
02 CHS
03 STO 6
04 RCL 0
05 RCL 7
06 x
07 g FRAC
08 STO 0
09 4
10 x
```

```
11 f INT
12 g x=0
13 GTO 31
14 1
15 −
16 g x=0
17 GTO 28
18 1
19 −
20 g x=0
21 GTO 26
22 RCL 6
23 STO 4
24 GTO 33
25 RCL 6
26 STO 3
27 GTO 33
28 RCL 6
29 STO 2
30 GTO 33
31 RCL 6
32 STO 1
33 1
34 STO +5
35 RCL 1
36 f PAUSE
37 RCL 4
38 f PAUSE
39 f x=y
40 GTO 47
41 RCL 2
42 f PAUSE
43 RCL 3
44 f PAUSE
45 f x≠y
46 GTO 01
47 RCL 5
48 GTO 00
```

EXERCISE 80

Before starting store x_0 in R_0. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
11 x
12 f INT
13 f PAUSE
14 STO x2
15 STO +2
16 RCL 2
17 k
18 f x≠y
19 GTO 01
20 RCL 1
21 R/S
22 0
23 STO 1
24 STO 2
25 GTO 01
```

EXERCISE 81

Before starting store x_0 in R_0. The first stop will display the mean and the second stop the standard deviation.

```
01  0
02  STO 1
03  STO 2
04  1
05  STO +1
06  RCL 0
07  1
08  4
09  7
10  x
11  g FRAC
12  STO 0
13  2
14  x
15  f INT
16  STO x2
17  STO +2
18  RCL 2
19  k
20  f x≠y
21  GTO 04
22  RCL 1
23  Σ +
24  RCL 3
25  f PAUSE
26  EEX
27  n
28  f x≠y
29  GTO 01
30  f x̄
31  R/S
32  f s
33  R/S
34  RCL 0
35  f REG
36  STO 0
37  GTO 01
```

EXERCISE 82

Before starting store x_0 in R_0 and p in R_7. The program can be repeated.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  RCL 7
11  +
12  f INT
13  f PAUSE
14  STO x2
15  STO +2
16  RCL 2
17  k
18  f x y
19  GTO 01
20  RCL 1
21  R/S
22  0
23  STO 1
24  STO 2
25  GTO 01
```

EXERCISE 83

Before starting store x_0 in R_0. The program can be repeated.

```
01  0
02  STO 1
03  STO 2
04  1
05  STO +1
06  RCL 0
07  1
08  4
09  7
10  x
11  g FRAC
12  STO 0
13  p
14  g NOP
15  +
16  f INT
17  STO x2
18  STO +2
19  RCL 2
20  k
21  f x≠y
22  GTO 04
23  RCL 1
24  Σ+
25  RCL 3
26  f PAUSE
27  EEX
28  n
29  f x≠y
30  GTO 01
31  f x̄
32  R/S
33  f s
34  R/S
35  RCL 0
36  f REG
37  STO 0
38  GTO 01
```

EXERCISE 84

Before starting store x_0 in R_0, k in R_3, N in R_4, and 147 in R_7. The program can be repeated.

```
01  1
02  STO +1
03  STO +2
04  RCL 0
05  RCL 7
06  x
07  g FRAC
08  STO 0
09  RCL 4
10  x
11  f INT
12  f PAUSE
13  STO 6
14  1
15  STO +1
16  STO +2
17  RCL 0
18  RCL 7
19  x
20  g FRAC
21  STO 0
22  RCL 4
23  x
24  f INT
25  f PAUSE
26  RCL 6
27  f x≠y
28  GTO 34
29  RCL 2
30  RCL 3
31  f x=y
32  GTO 39
33  GTO 14
34  ↓
35  STO 6
36  1
37  STO 2
38  GTO 14
```

```
39 RCL 1
40 R/S
41 0
42 STO 1
43 STO 2
44 GTO 01
```

EXERCISE 85

Before starting store x_0 in R_0, k in R_3, N in R_4, and 147 in R_7.

```
01 1
02 STO +1
03 STO +2
04 RCL 0
05 RCL 7
06 x
07 g FRAC
08 STO 0
09 RCL 4
10 x
11 f INT
12 STO 6
13 1
14 STO +1
15 STO +2
16 RCL 0
17 RCL 7
18 x
19 g FRAC
20 STO 0
21 RCL 4
22 x
23 f INT
24 RCL 6
25 f x≠y
26 GTO 32
27 RCL 2
28 RCL 3
29 f x=y
30 GTO 37
31 GTO 13
32 ↓
33 STO 6
34 1
35 STO 2
36 GTO 13
37 1
38 STO +5
39 RCL 5
40 f PAUSE
41 EEX
42 s
43 f x=y
44 GTO 48
45 0
46 STO 2
47 GTO 01
48 RCL 1
49 GTO 00
```

EXERCISE 86

Before starting store x_0 in R_0 and n in R_7. When the program stops, the number of runs is displayed. The program can be repeated.

```
01 1
02 STO 3
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
```

```
11 x
12 f INT
13 f PAUSE
14 STO 1
15 RCL 0
16 1
17 4
18 7
19 x
20 g FRAC
21 STO 0
22 2
23 x
24 f INT
25 f PAUSE
26 STO 2
27 RCL 1
28 f x=y
29 GTO 32
30 1
31 STO +3
32 1
33 STO +4
34 RCL 4
35 RCL 7
36 f x=y
37 GTO 40
38 RCL 2
39 GTO 14
40 RCL 3
41 R/S
42 0
43 STO 4
44 GTO 01
```

EXERCISE 87

Before starting store x_0 in R_0, n in R_6, and 147 in R_7. When the program stops, the total number of runs is displayed. Division by 10^s will give the average.

```
01 RCL 0
02 RCL 7
03 x
04 g FRAC
05 STO 0
06 2
07 x
08 f INT
09 STO 1
10 1
11 STO +4
12 RCL 0
13 RCL 7
14 x
15 g FRAC
16 STO 0
17 2
18 x
19 f INT
20 STO 2
21 RCL 1
22 f x=y
23 GTO 26
24 1
25 STO +3
26 1
27 STO +4
28 RCL 4
29 RCL 6
30 f x=y
31 GTO 35
32 RCL 2
33 STO 1
34 GTO 12
35 0
36 STO 4
37 1
38 STO +5
39 RCL 5
40 f PAUSE
```

```
41  EEX
42  s
43  f x≠y
44  GTO 01
45  STO +3
46  RCL 3
47  GTO 00
```

EXERCISE 88

The program in Exercise 86 can be used after the following changes. Before starting store x_0 in R_0, p in R_6, and n in R_7. Change program step 10 to "RCL 6," program step 11 to "+," program step 22 to "RCL 6," and finally program step 23 to "+." The program can be repeated.

EXERCISE 89

Before starting store x_0 in R_0, p in R_6, and 147 in R_7.

```
01  RCL 0
02  RCL 7
03  x
04  g FRAC
05  STO 0
06  RCL 6
07  +
08  f INT
09  STO 1
10  1
11  STO +4
12  RCL 0
13  RCL 7
14  x
15  g FRAC
16  STO 0
17  RCL 6
18  +
19  f INT
20  STO 2
21  RCL 1
22  f x=y
23  GTO 26
24  1
25  STO +3
26  1
27  STO +4
28  RCL 4
29  n
30  g NOP
31  f x=y
32  GTO 36
33  RCL 2
34  STO 1
35  GTO 12
36  0
37  STO 4
38  1
39  STO +5
40  RCL 5
41  f PAUSE
42  EEX
43  s
44  f x≠y
45  GTO 01
46  STO +3
47  RCL 3
48  GTO 00
```

EXERCISE 90

Before starting store x_0 in R_0, N in R_6, and 147 in R_7. When the program stops, the waiting time is displayed. The program can be repeated.

```
01  1
02  STO +2
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  RCL 6
09  x
10  f INT
11  f PAUSE
12  STO 1
13  1
14  STO +2
15  RCL 0
16  RCL 7
17  x
18  g FRAC
19  STO 0
20  RCL 6
21  x
22  f INT
23  f PAUSE
24  RCL 1
25  f x<y
26  GTO 30
27  ↓
28  STO 1
29  GTO 13
30  RCL 2
31  R/S
32  0
33  STO 2
34  GTO 01
```

EXERCISE 91

Before starting store x_0 in R_0.

```
01  1
02  STO +2
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  N
11  g NOP
12  x
13  f INT
14  STO 1
15  1
16  STO +2
17  RCL 0
18  1
19  4
20  7
21  x
22  g FRAC
23  STO 0
24  N
25  g NOP
26  x
27  f INT
28  RCL 1
29  f x<y
30  GTO 34
31  ↓
32  STO 1
33  GTO 15
34  RCL 2
```

```
35 Σ+
36 RCL 3
37 f PAUSE
38 EEX
39 s
40 f x=y
41 GTO 45
42 0
43 STO 2
44 GTO 01
45 f x̄
46 R/S
47 f s
48 R/S
49 GTO 00
```

EXERCISE 92

Before starting store x_0 in R_0, k in R_3, N in R_4, and 147 in R_7. The program makes 147 trials (guess why this number is used) of the experiment. When the program stops, recall R_1 and divide by 147 to obtain the mean $\overline{x}$.

```
01 1
02 STO +1
03 STO +2
04 RCL 0
05 RCL 7
06 x
07 g FRAC
08 STO 0
09 RCL 4
10 x
11 f INT
12 STO 6
13 1
14 STO +1
15 STO +2
16 RCL 0
17 RCL 7
18 x
19 g FRAC
20 STO 0
21 RCL 4
22 x
23 f INT
24 RCL 6
25 f x≥y
26 GTO 36
27 RCL 2
28 RCL 3
29 f x=y
30 GTO 41
31 ↓
32 ↓
33 ↓
34 STO 6
35 GTO 13
36 ↓
37 STO 6
38 1
39 STO 2
40 GTO 13
41 1
42 STO +5
43 RCL 5
44 RCL 7
45 f x=y
46 GTO 00
47 0
48 STO 2
49 GTO 01
```

EXERCISE 93

Before starting store x_0 in R_0, N in R_6, and 147 in R_7. When the program stops, divide the displayed number by 10^s to get the mean $\bar{x}$. The program can be repeated.

```
01  1
02  STO +3
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  RCL 6
09  x
10  f INT
11  STO 1
12  1
13  STO +3
14  RCL 0
15  RCL 7
16  x
17  g FRAC
18  STO 0
19  RCL 6
20  x
21  f INT
22  STO 2
23  1
24  —
25  RCL 1
26  f x=y
27  GTO 31
28  RCL 2
29  STO 1
30  GTO 12
31  1
32  STO +4
33  RCL 4
34  f PAUSE
35  EEX
36  s
37  f x≠y
38  GTO 01
39  RCL 3
40  R/S
41  0
42  STO 3
43  STO 4
44  GTO 01
```

EXERCISE 94

Change program step 23 in the preceding program to "k."

EXERCISE 95

The following program gives q_3, q_4, and so on at successive stops. The result 3.875 is interpreted as "q_3 = 0.875" and the result 20.21297169 is interpreted as "q_{20} = 0.21297169."

```
01  2
02  STO 5
03  1
04  STO 1
05  STO 2
06  STO 3
07  RCL 1
08  8
09  ÷
10  STO 4
11  RCL 2
12  4
```

```
13  ÷
14  STO +4
15  RCL 3
16  2
17  ÷
18  STO +4
19  1
20  STO +5
21  RCL 2
22  STO 1
23  RCL 3
24  STO 2
25  RCL 4
26  STO 3
27  RCL 5
28  +
29  R/S
30  GTO 07
```

EXERCISE 96

Before starting store x_0 in R_0, p in R_6, and 147 in R_7. When the program stops, the waiting time is displayed. The program can be repeated.

```
01  1
02  STO +2
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  RCL 6
09  +
10  f INT
11  f PAUSE
12  STO 1
13  1
14  STO +2
15  RCL 0
16  RCL 7
17  x
18  g FRAC
19  STO 0
20  RCL 6
21  +
22  f INT
23  f PAUSE
24  RCL 1
25  f x≠y
26  GTO 13
27  RCL 2
28  R/S
29  0
30  STO 2
31  GTO 01
```

EXERCISE 97

Before starting store x_0 in R_0, p in R_6, and 147 in R_7. When the program stops, divide by 10^n to obtain the mean $\overline{x}$. The program can be repeated.

```
01  1
02  STO +2
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  RCL 6
09  +
10  f INT
```

```
11 STO 1
12 1
13 STO +2
14 RCL 0
15 RCL 7
16 x
17 g FRAC
18 STO 0
19 RCL 6
20 +
21 f INT
22 RCL 1
23 f x≠y
24 GTO 12
25 1
26 STO +3
27 RCL 3
28 f PAUSE
29 EEX
30 n
31 f x≠y
32 GTO 01
33 RCL 2
34 R/S
35 0
36 STO 2
37 STO 3
38 GTO 01
```

EXERCISE 98

Before starting store x_0 in R_0, $-a$ in R_5, and b in R_6. The first stop will display the duration of the random walk; the second stop will display the end point of the walk. The program can be repeated.

```
01 1
02 STO +2
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
11 x
12 f INT
13 2
14 x
15 1
16 −
17 STO +1
18 RCL 1
19 f PAUSE
20 RCL 6
21 f x=y
22 GTO 28
23 RCL 1
24 RCL 5
25 f x=y
26 GTO 28
27 GTO 01
28 RCL 2
29 R/S
30 RCL 1
31 R/S
32 0
33 STO 1
34 STO 2
35 GTO 01
```

EXERCISE 99

Before starting store x_0 in R_0, $-a$ in R_5, and b in R_6. The first stop will display the frequency for endpoint b, and the second stop will display the total number of steps in all walks. Before repetition, clear $R_1 - R_4$.

```
01 1
02 STO +2
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
11 x
12 f INT
13 2
14 x
15 1
16 —
17 STO +1
18 RCL 1
19 RCL 6
20 f x=y
21 GTO 27
22 RCL 1
23 RCL 5
24 f x=y
25 GTO 27
26 GTO 01
27 RCL 1
28 g x 0
29 GTO 32
30 1
31 STO +3
32 1
33 STO +4
34 RCL 4
35 f PAUSE
36 EEX
37 n
38 f x=y
39 GTO 43
40 0
41 STO 1
42 GTO 01
43 RCL 3
44 R/S
45 RCL 2
46 R/S
47 GTO 00
```

EXERCISE 100

Before starting store x_0 in R_0. The program can be repeated.

```
01 1
02 STO 1
03 f PAUSE
04 STO 2
05 RCL 0
06 1
07 4
08 7
09 x
10 g FRAC
11 STO 0
12 2
13 x
14 f INT
15 2
16 x
17 1
18 STO +1
19 —
20 STO +2
21 RCL 2
22 f PAUSE
23 g x 0
24 GTO 05
```

25	RCL 1	28	STO 1
26	R/S	29	STO 2
27	0	30	GTO 01

EXERCISE 101

Before starting store x_0 in R_0. The first stop will display the mean $\overline{x}$ and the second the standard deviation s. The program can be repeated.

01	1	22	GTO 04
02	STO 1	23	RCL 1
03	STO 2	24	$\Sigma +$
04	RCL 0	25	RCL 3
05	1	26	*f* PAUSE
06	4	27	EEX
07	7	28	*n*
08	x	29	$f\, x=y$
09	*g* FRAC	30	GTO 35
10	STO 0	31	0
11	2	32	STO 1
12	x	33	STO 2
13	*f* INT	34	GTO 01
14	2	35	$f\,\overline{x}$
15	x	36	R/S
16	1	37	$f\, s$
17	STO +1	38	R/S
18	—	39	RCL 0
19	STO +2	40	*f* REG
20	RCL 2	41	STO 0
21	$g\, x \neq 0$	42	GTO 01

EXERCISE 102

Use the programs for Exercises 98 and 99 with the following changes. Store p in R_6. Change program step 10 to "RCL 6" and program step 11 to "+."

EXERCISE 103

Before starting store x_0 in R_0 and $2n$ in R_6. When the program stops, the number of returns to the origin is displayed. The program can be repeated.

```
01 1
02 STO +2
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
11 x
12 f INT
13 2
14 x
15 1
16 —
17 STO +1
18 RCL 1
19 f PAUSE
20 g x≠0
21 GTO 24
22 1
23 STO +3
24 RCL 2
25 RCL 6
26 f x≠y
27 GTO 01
28 RCL 3
29 R/S
30 0
31 STO 1
32 STO 2
33 STO 3
34 GTO 01
```

EXERCISE 104

Before starting store x_0 in R_0 and $2n$ in R_6. When the program stops, divide by 10^s to obtain the mean $\overline{x}$.

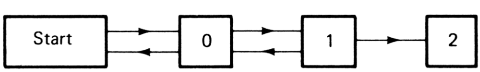
```
01 1
02 STO +2
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
11 x
12 f INT
13 2
14 x
15 1
16 —
17 STO +1
18 RCL 1
19 g x≠0
20 GTO 23
21 1
22 STO +3
23 RCL 2
24 RCL 6
25 f x≠y
26 GTO 01
27 1
28 STO +4
29 RCL 4
30 f PAUSE
31 EEX
32 s
33 f x=y
34 GTO 39
35 0
36 STO 1
37 STO 2
38 GTO 01
39 RCL 3
40 R/S
```

EXERCISE 105

The following program uses the first method described in the commentary to this exercise. Before starting store x_0 in R_0; -1 in R_1, R_2, R_3, and R_4; and 147 in R_7. When the program stops, the duration is displayed. The program can be repeated.

```
01  1
02  STO +5
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  3
09  x
10  f INT
11  g x=0
12  GTO 23
13  1
14  −
15  g x=0
16  GTO 20
17  RCL 4
18  STO x3
19  GTO 25
20  RCL 4
21  STO x2
22  GTO 25
23  RCL 4
24  STO x1
25  RCL 1
26  f PAUSE
27  RCL 2
28  f PAUSE
29  RCL 3
30  f PAUSE
31  f x≠y
32  GTO 01
33  RCL 1
34  f x≠y
35  GTO 01
36  g x<0
37  GTO 01
38  RCL 5
39  R/S
40  RCL 4
41  STO x1
42  STO x2
43  STO x3
44  0
45  STO 5
46  GTO 01
```

The following program uses the second method described in the exercise commentary. Register R_1 records the states of the walk, which are numbered according to the figure below. These states are indicated. Returns to the origin are not indicated. Before



```
01 1
02 STO +2
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 3
11 x
12 f INT
13 STO 6
14 RCL 1
15 f PAUSE
16 g x≠0
17 GTO 27
18 RCL 6
19 g x=0
20 GTO 24
21 1
22 STO 1
23 GTO 01
24 2
25 STO + 2
26 GTO 03
27 RCL 6
28 g x≠0
29 GTO 40
30 1
31 STO +2
32 2
33 f PAUSE
34 RCL 2
35 R/S
36 0
37 STO 1
38 STO 2
39 GTO 01
40 0
41 STO 1
42 GTO 01
```

EXERCISE 106

The following program uses the first method in the commentary to Exercise 105. Before starting store x_0 in R_0; -1 in R_1, R_2, R_3, and R_4; and 147 in R_7. When the program stops, divide by 10^n to obtain the average duration $\overline{x}$.

```
01 1
02 STO +5
03 RCL 0
04 RCL 7
05 x
06 g FRAC
07 STO 0
08 3
09 x
10 f INT
11 g x=0
12 GTO 23
13 1
14 −
15 g x=0
16 GTO 20
17 RCL 4
18 STO x3
19 GTO 25
20 RCL 4
21 STO x2
22 GTO 25
23 RCL 4
24 STO x1
25 RCL 1
26 RCL 2
27 f x≠y
28 GTO 01
```

```
29 RCL 3
30 f x≠y
31 GTO 01
32 g x<0
33 GTO 01
34 1
35 STO +6
36 RCL 6
37 f PAUSE
38 EEX
39 n
40 f x=y
41 GTO 47
42 RCL 4
43 STO x1
44 STO x2
45 STO x3
46 GTO 01
47 RCL 5
48 GTO 00
```

EXERCISE 107

The following program uses the first method in the commentary to Exercise 105. Before starting store x_0 in R_0; -1 in R_1, R_2, R_3, R_4 and R_5; and 147 in R_7. When the program stops, the duration of the walk can be recalled from register R_6. Before repetition, store -1 in R_1, R_2, R_3, and R_4 and clear R_6.

```
01 1
02 STO +6
03 RCL 0
04 RCL 7
05 x
06 g FRAC
07 STO 0
08 4
09 x
10 f INT
11 g x=0
12 GTO 30
13 1
14 −
15 g x=0
16 GTO 27
17 1
18 −
19 g x=0
20 GTO 24
21 RCL 5
22 STO x4
23 GTO 32
24 RCL 5
25 STO x3
26 GTO 32
27 RCL 5
28 STO x2
29 GTO 32
30 RCL 5
31 STO x1
32 RCL 1
33 f PAUSE
34 RCL 2
35 f PAUSE
36 RCL 3
37 f PAUSE
38 RCL 4
39 f PAUSE
40 f x≠y
41 GTO 01
42 RCL 2
43 f x≠y
44 GTO 01
45 RCL 1
46 f x≠y
47 GTO 01
48 g x<0
49 GTO 01
```

EXERCISE 109

In the following program register R_2 keeps track of the states of the walk. The states are numbered according to the figure in the commentary. Before starting store x_0 in R_0 and 147 in R_7. When the program stops, the duration is displayed. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 RCL 7
05 x
06 g FRAC
07 STO 0
08 2
09 x
10 f INT
11 STO 6
12 RCL 2
13 f PAUSE
14 g x=0
15 GTO 23
16 1
17 –
18 g x=0
19 GTO 29
20 1
21 STO 2
22 GTO 01
23 RCL 6
24 g x=0
25 GTO 01
26 1
27 STO 2
28 GTO 01
29 RCL 6
30 g x=0
31 GTO 35
32 2
33 STO 2
34 GTO 01
35 3
36 f PAUSE
37 1
38 STO +1
39 RCL 1
40 R/S
41 0
42 STO 1
43 STO 2
44 GTO 01
```

EXERCISE 110

Use the program from Exercise 109 with the following changes. Change program step 13 to "*g* NOP" and use the following program steps from step 35 on:

```
35 1
36 STO +1
37 STO +3
38 RCL 3
39 f PAUSE
40 EEX
41 n
42 f x=y
43 GTO 47
44 0
45 STO 2
46 GTO 01
47 RCL 1
48 GTO 00
```

Tony Elmroth, a student at Chalmers University of Technology, has written a program that treats the corresponding random walk on a cube. Claes Löfgren, another student at Chalmers, has calculated the expectation in that case to 28389/1330.

EXERCISE 111

Before starting store x_0 in R_0, 1.1111111 in R_1, 1.1111118 in R_2, and 8.1111111 in R_3. The program can be repeated.

```
01  f SCI 7
02  7
03  STO 5
04  1
05  STO 4
06  GTO 32
07  RCL 4
08  ÷
09  RCL 2
10  f x=y
11  GTO 43
12  CLX
13  RCL 3
14  f x=y
15  GTO 32
16  RCL 0
17  1
18  4
19  7
20  x
21  g FRAC
22  STO 0
23  2
24  x
25  f INT
26  g x=0
27  GTO 32
28  1
29  0
30  STO x5
31  GTO 35
32  1
33  0
34  STO ÷5
35  STO x4
36  RCL 5
37  RCL 1
38  +
39  RCL 4
40  x
41  f PAUSE
42  GTO 07
43  RCL 4
44  f log
45  f FIX 0
46  GTO 00
```

This program was written by the author's son Johan, 11 years old.

EXERCISE 112

Before starting store x_0 in R_0 and n in R_6. The first stop will display $\hat{p}$ and the second stop p. The program can be repeated.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
```

```
09 STO 0
10 RCL 1
11 9
12 f x≠y
13 GTO 01
14 RCL 0
15 STO 2
16 1
17 STO +3
18 RCL 0
19 1
20 4
21 7
22 x
23 g FRAC
24 STO 0
25 RCL 2
26 +
27 f INT
28 STO +4
29 RCL 3
30 f PAUSE
31 RCL 6
32 f x≠y
33 GTO 16
34 RCL 4
35 RCL 6
36 ÷
37 R/S
38 RCL 2
39 R/S
40 0
41 STO 1
42 STO 3
43 STO 4
44 GTO 01
```

EXERCISE 113

Before starting store x_0 in R_0, n in R_6, and 147 in R_7. The first stop will display $\hat{N}_1$ and the second stop N. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 RCL 7
05 x
06 g FRAC
07 STO 0
08 RCL 1
09 9
10 f x≠y
11 GTO 01
12 RCL 0
13 EEX
14 3
15 x
16 f INT
17 STO 2
18 1
19 STO +3
20 RCL 0
21 RCL 7
22 x
23 g FRAC
24 STO 0
25 RCL 2
26 x
27 f INT
28 1
29 +
30 f PAUSE
31 STO +4
32 RCL 3
33 RCL 6
34 f x≠y
35 GTO 18
36 RCL 4
37 RCL 6
38 ÷
```

39	2	44	0
40	x	45	STO 1
41	R/S	46	STO 3
42	RCL 2	47	STO 4
43	R/S	48	GTO 01

EXERCISE 114

Before starting store x_0 in R_0, n in R_6, and 147 in R_7. The first stop will display $\hat{N}_2$ and the second stop N. Clear R_1, R_3, and R_4 before the program is repeated.

01	1	25	RCL 2
02	STO +1	26	x
03	RCL 0	27	f INT
04	RCL 7	28	1
05	x	29	+
06	g FRAC	30	f PAUSE
07	STO 0	31	RCL 4
08	RCL 1	32	$f\ x \geqslant y$
09	9	33	GTO 36
10	$f\ x \neq y$	34	↓
11	GTO 01	35	STO 4
12	RCL 0	36	RCL 3
13	EEX	37	RCL 6
14	3	38	$f\ x \neq y$
15	x	39	GTO 18
16	f INT	40	1
17	STO 2	41	RCL 6
18	1	42	÷
19	STO +3	43	1
20	RCL 0	44	+
21	RCL 7	45	RCL 4
22	x	46	x
23	g FRAC	47	R/S
24	STO 0	48	RCL 2
		49	GTO 00

EXERCISE 115

Before starting store x_0 in R_0 and p in R_7.

01	1	04	1
02	STO +1	05	4
03	RCL 0	06	7

```
07  x
08  g FRAC
09  STO 0
10  RCL 7
11  +
12  f INT
13  STO +2
14  RCL 2
15  RCL 1
16  ÷
17  f PAUSE
18  GTO 01
```

EXERCISE 116

Before starting store x_0 in R_0 and N in R_1.

```
01  RCL 0
02  1
03  4
04  7
05  x
06  g FRAC
07  STO 0
08  RCL 1
09  x
10  f INT
11  1
12  +
13  Σ+
14  f x̄
15  2
16  x
17  1
18  −
19  f PAUSE
20  GTO 01
```

EXERCISE 117

Before starting store x_0 in R_0 and N in R_7.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  RCL 7
11  x
12  f INT
13  1
14  +
15  RCL 2
16  f x≥y
17  GTO 20
18  ↓
19  STO 2
20  1
21  RCL 1
22  ÷
23  1
24  +
25  RCL 2
26  x
27  f PAUSE
28  GTO 01
```

EXERCISE 118

Before starting store x_0 in R_0. This program spins an N-spinner with $N = 100$ ten times and calculates $2\overline{x}$. This is repeated ten times and the variance of the ten estimates $2\overline{x}$ is calculated. The program can be repeated.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  1
11  0
12  0
13  x
14  f INT
15  1
16  +
17  STO +2
18  RCL 1
19  1
20  0
21  f x≠y
22  GTO 01
23  RCL 2
24  1
25  0
26  ÷
27  2
28  x
29  Σ+
30  f PAUSE
31  ENTER
32  1
33  0
34  f x=y
35  GTO 40
36  0
37  STO 1
38  STO 2
39  GTO 01
40  f s
41  g x²
42  R/S
43  RCL 0
44  f REG
45  STO 0
46  GTO 01
```

The following program is similar to the preceding program but here N is estimated by $\frac{n+1}{n}x_{\max}$.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  1
11  0
12  0
13  x
14  f INT
15  1
16  +
17  RCL 2
18  f x≥y
19  GTO 22
20  ↓
21  STO 2
22  RCL 1
23  1
24  0
25  f x≠y
26  GTO 01
```

```
27  1
28  •
29  1
30  RCL 2
31  x
32  Σ +
33  f PAUSE
34  ENTER
35  1
36  0
37  f x=y
38  GTO 43
39  0
40  STO 1
41  STO 2
42  GTO 01
43  f s
44  g x²
45  R/S
46  RCL 0
47  f REG
48  STO 0
49  GTO 01
```

EXERCISE 119

Before starting store x_0 in R_0, p in R_7 and n in R_6. When the program stops, the number of times the interval contains p is stored in R_4. In this case the random digits are generated by the 83-generator.

```
01  1
02  STO +1
03  RCL 0
04  8
05  3
06  x
07  g FRAC
08  STO 0
09  RCL 7
10  +
11  f INT
12  STO +2
13  RCL 1
14  RCL 6
15  f x≠y
16  GTO 01
17  RCL 2
18  x⇌y
19  ÷
20  STO 3
21  CHS
22  1
23  STO +5
24  +
25  RCL 3
26  x
27  RCL 6
28  ÷
29  f √x
30  2
31  x
32  RCL 7
33  RCL 3
34  −
35  g ABS
36  f x≥y
37  GTO 40
38  1
39  STO +4
40  RCL 5
41  f PAUSE
42  EEX
43  m
44  f x=y
45  GTO 00
46  0
47  STO 1
48  STO 2
49  GTO 01
```

EXERCISE 120

Before starting store x_0 in R_0, n in R_6, and 147 in R_7. The first stop will display the point estimate $\hat{p}$ of p, the second stop will display $2\sqrt{\hat{p}(1-\hat{p})/n}$, and the third stop will display the probability p. Clear R_1, R_3, and R_4 before the program is repeated.

```
01  1
02  STO +1
03  RCL 0
04  RCL 7
05  x
06  g FRAC
07  STO 0
08  RCL 1
09  9
10  f x≠y
11  GTO 01
12  RCL 0
13  STO 2
14  1
15  STO +3
16  RCL 0
17  RCL 7
18  x
19  g FRAC
20  STO 0
21  RCL 2
22  +
23  f INT
24  STO +4
25  RCL 3
26  f PAUSE
27  RCL 6
28  f x≠y
29  GTO 14
30  RCL 4
31  RCL 6
32  ÷
33  R/S
34  STO 5
35  CHS
36  1
37  +
38  RCL 5
39  x
40  RCL 6
41  ÷
42  f √x
43  2
44  x
45  R/S
46  RCL 2
47  GTO 00
```

EXERCISE 121

Before starting store x_0 in R_0. The first stop will display the total number of children. Divide by 10^n to get the average family size. The second stop will display the sex proportion. The program can be repeated.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  2
11  x
12  f INT
```

```
13  STO +2
14  STO 3
15  1
16  STO +1
17  RCL 0
18  1
19  4
20  7
21  x
22  g FRAC
23  STO 0
24  2
25  x
26  f INT
27  STO +2
28  RCL 3
29  f x=y
30  GTO 15
31  1
32  STO +4
33  RCL 4
34  f PAUSE
35  EEX
36  n
37  f x≠y
38  GTO 01
39  RCL 1
40  R/S
41  RCL 2
42  x⇌y
43  ÷
44  R/S
45  RCL 0
46  f REG
47  STO 0
48  GTO 01
```

EXERCISE 122

Before starting store x_0 in R_0. The first stop displays the total number of children and the second the sex proportion. The program can be repeated.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  2
11  x
12  f INT
13  STO +2
14  g x=0
15  GTO 01
16  1
17  STO +3
18  RCL 3
19  f PAUSE
20  EEX
21  n
22  f x≠y
23  GTO 01
24  RCL 1
25  R/S
26  RCL 2
27  x↔y
28  ÷
29  R/S
30  RCL 0
31  f REG
32  STO 0
33  GTO 01
```

EXERCISE 123

Before starting store x_0 in R_0. The first stop displays the total number of children and the second the sex proportion. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
11 x
12 f INT
13 STO +2
14 g x≠0
15 GTO 20
16 RCL 1
17 3
18 f x≠y
19 GTO 01
20 1
21 STO +3
22 RCL 3
23 f PAUSE
24 EEX
25 n
26 f x≠y
27 GTO 01
28 RCL 1
29 R/S
30 RCL 2
31 x↔y
32 ÷
33 R/S
34 RCL 0
35 f REG
36 STO 0
37 GTO 01
```

EXERCISE 124

Before starting store x_0 in R_0. The first stop displays the total number of children and the second the sex proportion. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 2
11 x
12 f INT
13 STO +2
14 STO +3
15 RCL 2
16 2
17 f x=y
18 GTO 23
19 RCL 1
20 5
21 f x≠y
22 GTO 01
23 1
24 STO +4
25 RCL 4
26 f PAUSE
```

```
27 EEX
28 n
29 f x=y
30 GTO 34
31 0
32 STO 2
33 GTO 01
34 RCL 1
35 R/S
36 RCL 3
37 x↔y
38 ÷
39 R/S
40 RCL 0
41 f REG
42 STO 0
43 GTO 01
```

EXERCISE 125

If the probability 0.48 for a girl is stored in a register, the previous programs can be used after a few obvious changes.

EXERCISE 126

Before starting store x_0 in R_0, p in R_6, and 147 in R_7. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 RCL 7
05 x
06 g FRAC
07 STO 0
08 RCL 6
09 +
10 f INT
11 f PAUSE
12 g x=0
13 GTO 01
14 STO +1
15 RCL 0
16 RCL 7
17 x
18 g FRAC
19 STO 0
20 RCL 6
21 +
22 f INT
23 f PAUSE
24 g x≠0
25 GTO 14
26 1
27 STO +1
28 RCL 0
29 RCL 7
30 x
31 g FRAC
32 STO 0
33 RCL 6
34 +
35 f INT
36 f PAUSE
37 g x=0
38 GTO 01
39 RCL 1
40 R/S
41 0
42 STO 1
43 GTO 01
```

EXERCISE 127

Before starting store x_0 in R_0, p in R_6, and 147 in R_7. When the program stops, divide by 10^n to get the average waiting time. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 RCL 7
05 x
06 g FRAC
07 STO 0
08 RCL 6
09 +
10 f INT
11 g x=0
12 GTO 01
13 STO +1
14 RCL 0
15 RCL 7
16 x
17 g FRAC
18 STO 0
19 RCL 6
20 +
21 f INT
22 g x≠0
23 GTO 13
24 1
25 STO +1
26 RCL 0
27 RCL 7
28 x
29 g FRAC
30 STO 0
31 RCL 6
32 +
33 f INT
34 g x=0
35 GTO 01
36 1
37 STO +2
38 RCL 2
39 f PAUSE
40 EEX
41 n
42 f x≠y
43 GTO 01
44 RCL 1
45 R/S
46 0
47 STO 1
48 STO 2
49 GTO 01
```

EXERCISE 128

Before starting store x_0 in R_0, −1 in R_2 (to make the first digit a record), and n in R_7. When the program stops, the number of records is displayed. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 1
11 0
12 x
13 f INT
14 f PAUSE
```

```
15 RCL 2
16 f x≥y
17 GTO 22
18 ↓
19 STO 2
20 1
21 STO +3
22 RCL 1
23 RCL 7
24 f x≠y
25 GTO 01
26 RCL 3
27 R/S
28 0
29 STO 1
30 STO 3
31 1
32 CHS
33 STO 2
34 GTO 01
```

EXERCISE 129

Before starting store x_0 in R_0, -1 in R_2, and n in R_7. When the program stops, divide by 10^m to get the average number of records. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 1
11 0
12 x
13 f INT
14 RCL 2
15 f x≥y
16 GTO 21
17 ↓
18 STO 2
19 1
20 STO +3
21 RCL 1
22 RCL 7
23 f x≠y
24 GTO 01
25 1
26 STO +4
27 RCL 4
28 f PAUSE
29 EEX
30 m
31 f x=y
32 GTO 39
33 0
34 STO 1
35 1
36 CHS
37 STO 2
38 GTO 01
39 RCL 3
40 R/S
41 0
42 STO 1
43 STO 3
44 STO 4
45 1
46 CHS
47 STO 2
48 GTO 01
```

EXERCISE 130

Before starting store x_0 in R_0, -1 in R_2, 147 in R_5, n in R_6, and N in R_7. When the program stops, divide by 10^m to get the relative frequency. Store -1 in R_2 and clear R_1, R_3, and R_4 before the program is repeated.

```
01  1
02  STO +1
03  RCL 0
04  RCL 5
05  x
06  g FRAC
07  STO 0
08  RCL 7
09  x
10  f INT
11  RCL 2
12  f x≥y
13  GTO 16
14  ↓
15  STO 2
16  RCL 6
17  1
18  −
19  RCL 1
20  f x≠y
21  GTO 01
22  RCL 0
23  RCL 5
24  x
25  g FRAC
26  STO 0
27  RCL 7
28  x
29  f INT
30  RCL 2
31  f x≥y
32  GTO 35
33  1
34  STO +3
35  1
36  STO +4
37  RCL 4
38  f PAUSE
39  EEX
40  m
41  f x=y
42  GTO 49
43  0
44  STO 1
45  1
46  CHS
47  STO 2
48  GTO 01
49  RCL 3
```

EXERCISE 131

Before starting store 1 in R_1 and R_5, and N in R_6 and R_7. The first stop displays n and the second $P(N,n)$.

```
01  1
02  STO +1
03  RCL 6
04  1
05  −
06  STO 6
07  RCL 7
08  ÷
09  RCL 5
10  x
11  STO 5
12  CHS
13  1
14  +
15  RCL 1
16  R/S
17  ↓
18  R/S
19  GTO 01
```

EXERCISE 132

Before starting store 1 in R_1 and R_5, and N in R_6 and R_7. The first stop displays N and the second n such that $P(N,n) \geqslant 0.5$.

```
01  1
02  STO +1
03  RCL 6
04  1
05  −
06  STO 6
07  RCL 7
08  ÷
09  RCL 5
10  x
11  STO 5
12  CHS
13  1
14  +
15  •
16  5
17  x↔y
18  f x<y
19  GTO 01
20  RCL 7
21  R/S
22  RCL 1
23  GTO 00
```

EXERCISE 133

```
01  3
02  6
03  4
04  ENTER
05  3
06  6
07  5
08  ÷
09  STO 2
10  1
11  STO +1
12  RCL 2
13  RCL 1
14  f y^x
15  CHS
16  1
17  +
18  f PAUSE
19  •
20  5
21  x↔y
22  f x<y
23  GTO 10
24  R/S
25  RCL 1
26  GTO 00
```

The program gives $p_{253} = 0.500477148$.

EXERCISE 134

Before starting store x_0 in R_0, a in R_1, and b in R_2. The successive numbers of votes for A and B are displayed.

```
01  RCL 1
02  STO 6
03  RCL 2
04  STO 7
05  RCL 0
06  1
07  4
08  7
09  x
10  g FRAC
11  STO 0
12  RCL 1
```

```
13  RCL 2
14  +
15  RCL 2
16  x↔y
17  ÷
18  RCL 0
19  +
20  f INT
21  g x=0
22  GTO 26
23  STO −2
24  STO +4
25  GTO 29
26  1
27  STO −1
28  STO +3
29  RCL 3
30  f PAUSE
31  RCL 4
32  f PAUSE
33  RCL 1
34  RCL 2
35  +
36  g x≠0
37  GTO 05
38  R/S
39  0
40  STO 3
41  STO 4
42  RCL 6
43  STO 1
44  RCL 7
45  STO 2
46  GTO 05
```

EXERCISE 135

Before starting store x_0 in R_0, a in R_1, and b in R_2. Store a in R_1 and b in R_2 before the program is repeated.

```
01  RCL 0
02  1
03  4
04  7
05  x
06  g FRAC
07  STO 0
08  RCL 1
09  RCL 2
10  +
11  RCL 2
12  x↔y
13  ÷
14  RCL 0
15  +
16  f INT
17  g x=0
18  GTO 22
19  STO −2
20  STO +4
21  GTO 25
22  1
23  STO −1
24  STO +3
25  RCL 3
26  f PAUSE
27  RCL 4
28  f PAUSE
29  f x≠y
30  GTO 33
31  1
32  STO +5
33  RCL 1
34  RCL 2
35  +
36  g x≠0
37  GTO 01
38  RCL 5
39  R/S
40  0
41  STO 3
42  STO 4
43  STO 5
44  GTO 01
```

EXERCISE 136

Before starting store x_0 in R_0 and 147 in R_7. When the program stops, recall R_5 and divide by 10^n to get the mean $\overline{x}$. Clear R_5 and R_6 before the program is repeated.

01 0
02 STO 3
03 STO 4
04 a
05 g NOP
06 STO 1
07 b
08 g NOP
09 STO 2
10 RCL 0
11 RCL 7
12 x
13 g FRAC
14 STO 0
15 RCL 2
16 RCL 1
17 RCL 2
18 +
19 ÷
20 RCL 0
21 +
22 f INT
23 $g\ x=0$
24 GTO 28
25 STO −2
26 STO +4
27 GTO 31
28 1
29 STO −1
30 STO +3
31 RCL 3
32 RCL 4
33 $f\ x\neq y$
34 GTO 37
35 1
36 STO +5
37 RCL 1
38 RCL 2
39 +
40 $g\ x\neq 0$
41 GTO 10
42 1
43 STO +6
44 RCL 6
45 f PAUSE
46 EEX
47 n
48 $f\ x\neq y$
49 GTO 01

EXERCISE 137

Before starting store x_0 in R_0 and 147 in R_7. When the program stops, recall R_5 and divide by 10^n to estimate the probability that A will *not* have more votes than B during the counting of votes. Clear R_5 and R_6 before the program is repeated.

01 0
02 STO 3
03 STO 4
04 a
05 g NOP
06 STO 1
07 b
08 g NOP
09 STO 2
10 RCL 0
11 RCL 7
12 x

```
13 g FRAC
14 STO 0
15 RCL 2
16 RCL 1
17 RCL 2
18 +
19 ÷
20 RCL 0
21 +
22 f INT
23 g x=0
24 GTO 28
25 STO −2
26 STO +4
27 GTO 31
28 1
29 STO −1
30 STO +3
31 RCL 3
32 RCL 4
33 f x≥y
34 GTO 41
35 RCL 1
36 RCL 2
37 +
38 g x≠0
39 GTO 10
40 GTO 43
41 1
42 STO +5
43 1
44 STO +6
45 RCL 6
46 EEX
47 n
48 f x≠y
49 GTO 01
```

EXERCISE 138

Before starting store x_0 in R_0, p in R_5, p_1 in R_6, and 147 in R_7. The first stop displays the time epoch and the second stop the number of customers in the system.

```
01 1
02 STO +1
03 RCL 0
04 RCL 7
05 x
06 g FRAC
07 STO 0
08 RCL 5
09 +
10 f INT
11 STO +2
12 RCL 1
13 R/S
14 RCL 2
15 R/S
16 g x=0
17 GTO 01
18 1
19 STO +1
20 RCL 0
21 RCL 7
22 x
23 g FRAC
24 STO 0
25 RCL 5
26 +
27 f INT
28 STO +2
29 RCL 0
30 RCL 7
31 x
32 g FRAC
33 STO 0
34 RCL 6
35 +
36 f INT
```

```
37 STO −2
38 RCL 1
39 R/S
40 RCL 2
41 R/S
42 g x=0
43 GTO 01
44 GTO 18
```

EXERCISE 139

Before starting store x_0 in R_0, p in R_5, p_1 in R_6, and 147 in R_7. The first stop will display the length of the busy period and the second the number of customers served. The program can be repeated.

```
01 1
02 STO 2
03 STO 3
04 1
05 STO +1
06 RCL 0
07 RCL 7
08 x
09 g FRAC
10 STO 0
11 RCL 5
12 +
13 f INT
14 STO +2
15 STO +3
16 RCL 0
17 RCL 7
18 x
19 g FRAC
20 STO 0
21 RCL 6
22 +
23 f INT
24 STO −2
25 RCL 2
26 f PAUSE
27 g x≠0
28 GTO 04
29 RCL 1
30 R/S
31 RCL 3
32 R/S
33 0
34 STO 1
35 GTO 01
```

EXERCISE 140

Before starting store x_0 in R_0, p in R_5, p_1 in R_6, and 147 in R_7. At the first stop divide by 10^n to get the average length of the busy periods; at the second stop divide by 10^n to get the average number of customers served during a busy period. The program can be repeated.

```
01 1
02 STO 2
03 STO +3
04 1
05 STO +1
06 RCL 0
07 RCL 7
08 x
09 g FRAC
10 STO 0
11 RCL 5
12 +
```

```
13  f INT
14  STO +2
15  STO +3
16  RCL 0
17  RCL 7
18  x
19  g FRAC
20  STO 0
21  RCL 6
22  +
23  f INT
24  STO −2
25  RCL 2
26  g x≠0
27  GTO 04
28  1
29  STO +4
30  RCL 4
31  f PAUSE
32  EEX
33  n
34  f x≠y
35  GTO 01
36  RCL 1
37  R/S
38  RCL 3
39  R/S
40  0
41  STO 1
42  STO 3
43  STO 4
44  GTO 01
```

EXERCISE 141

Before starting store x_0 in R_0, p in R_6, and r in R_7. The program can be repeated.

```
01  1
02  STO +1
03  RCL 0
04  1
05  4
06  7
07  x
08  g FRAC
09  STO 0
10  RCL 6
11  +
12  f INT
13  f PAUSE
14  STO +2
15  RCL 2
16  RCL 7
17  f x≠y
18  GTO 01
19  RCL 2
20  RCL 1
21  ÷
22  R/S
23  0
24  STO 1
25  STO 2
26  GTO 01
```

EXERCISE 142

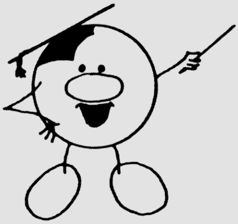
Before starting store x_0 in R_0, r in R_6, and p in R_7. The program can be repeated.

```
01 1
02 STO +1
03 RCL 0
04 1
05 4
06 7
07 x
08 g FRAC
09 STO 0
10 RCL 7
11 +
12 f INT
13 STO +2
14 RCL 2
15 RCL 6
16 f x≠y
17 GTO 01
18 RCL 2
19 g NOP
20 g NOP
21 RCL 1
22 g NOP
23 g NOP
24 ÷
25 STO +3
26 1
27 STO +4
28 RCL 4
29 f PAUSE
30 EEX
31 n
32 f x=y
33 GTO 38
34 0
35 STO 1
36 STO 2
37 GTO 01
38 RCL 3
39 RCL 4
40 ÷
41 R/S
42 0
43 STO 1
44 STO 2
45 STO 3
46 STO 4
47 GTO 01
```

EXERCISE 143

Use the program for Exercise 142 with the following new steps.

```
19 1
20 −
22 1
23 −
```



Lennart Råde wrote this book to help you to discover the world of probability with your programmable calculator. You will need NO previous experience either in probability theory or in programming to learn both from *Take A Chance With Your Calculator*.

Professor Råde was born in Ramsele, Sweden, and teaches statistical math at Chalmers University of Technology in Göteborg. He has written more articles than we can begin to mention in this small space. This is his thirteenth book–a lucky number for **dilithium Press** and for you!

Books from dilithium Press

Home Computers: 2^{10} Questions & Answers
Volume 1: Hardware
Rich Didday
ISBN: 0-918398-00-2

Home Computers: 2^{10} Questions & Answers
Volume 2: Software
Rich Didday
ISBN: 0-918398-01-0

Beginning BASIC
Paul Chirlian
ISBN: 0-918398-06-1

Step by Step Introduction to 8080 Micro-processor Systems
James Melsa & David Cohn
ISBN: 0-918398-04-5

ISBN: 0-918398-07-X

dilithium PRESS
P. O. Box 92
Forest Grove, Oregon 97116